School of Computing - Advanced Technologies
University of Abertay Dundee, UK

Vision based Interactive Toys Environment

By

Eiman Kanjo

A thesis presented in fulfilment of the requirements for
the degree of Doctor of Philosophy

March-2005

I certify that this thesis is the true and accurate version of the thesis approved by the examiners.

Abstract

Recently, attention has focused on the development of computer-user interfaces, which combine digital information with physical environments. In this thesis, the author assumed the hypothesis that Computer Vision is a technique that has great potential to support the concept of marrying the digital world with the physical one. This was proven by developing a novel sensing technology that allows users to use their home computer and an ordinary web camera to detect multiple small physical objects, referred to as "Interactive Toys" (e.g. any pieces of pawns, cars, animals) in a collaborative environment, referred to as "Interactive Toys Environment" (e.g. playsets or boardgames).

The study comprises the development of novel algorithms that detect the intrusion of hands into the camera's view by monitoring users' actions ("move and place"). When the user finishes performing the current task by withdrawing their hands from the camera's view, the system automatically responds according to the positions, directions and colours of the moved objects or toys. Toys tracking and identification are based on novel change detection algorithms that allow the system to know which toy or toys have moved in the last turn. Moreover, particular efforts were given to map the positions of the toys in the Interactive Toys Environment to the image coordinates system under various working conditions (e.g. camera vibration or background slightly moved).

Usability and effectiveness of the Interactive Toys Environment are demonstrated through a number of applications of computer games used and played by children. Some of these applications were evaluated in user studies, which have shown that Interactive Toys Environment provides a new application of computer vision to facilitate natural human-computer interaction (e.g. without forcing users to wear special equipment such as special sensing gloves or using chipped-up or tagged toys). It also encourages active thinking and social communication and improves the interaction between users and computers.

Acknowledgement

Praise and thanks must be given first to Allah who has provided me with health patience courage and knowledge to complete this study.

I would like to thank my supervisor Prof. Peter Astheimer for having the confidence in my abilities to allow me to pursue the topics which excite me. Peter has been a constant source of support and motivation throughout the last four years of my association with the ICCAVE.

In addition, this thesis has been immeasurably improved by the thorough reading and thoughtful feedback of my second supervisor Prof. David Bradley.

The current and past members of ICCAVE have made the lab energetic and fun environment to spend the last years. Particular thanks go to the ICCAVE manager Penny Robertson, for her time and effort to overcome of many administrative works. I would also like to thank my husband Omar for his support whilst I was grapple with this dissertation and I certainly would not have completed this work—or done a great many other notable things—without him. I am also indebted to his constant help in pursuing a research and his great effort in helping me looking after our precious daughters aged one and four.

I would like to thank family in Syria, especially my parents, for being a great inspiration for my work and for their constant support and help to achieve my goals in this life.

Declaration

Elements from this thesis have appeared in the following publications:

2004- April	Kanjo E., Astheimer P. and Marshall I., "Supporting Face-to-Face Communications through Interactive Toys Space," In Proc. of IEEE Intl. Conf. on Information & Communication Technologies ICCTA'04 conference, Damascus, pp. 523- 524.
2004-April	Lu K. T., Astheimer P., Kanjo E.,(2004), "Vision-Based Innovative Network Games," In Proc. of Intl. conf. Application and Development of Computer Games, ADCOG'04, Hong Kong, China, pp. 34-43.
2003- Dec	Kanjo E. and Astheimer P., (2003) "Computer Vision for Human Computer Interaction". Poster presentation at the House of Commons: British Younger Engineers, London.
2003- Oct	Kanjo E., Ren J., Astheimer P. and Marshall I., "A General Framework for Vision-Based Interactive Board Games", Games-On conference 2003, London.
2002- Oct	Kanjo E., Astheimer P., (2003), "Interactive Environment for Storytelling by Playmates Toys," In Proc. of ACM, Collaborative Virtual Environments, Storytelling Workshop, Bonn, Germany, SIGGROUP Bulletin. ACM, NY, vol.23, no.2, pp.6-7.
2002- Sep	Kanjo E. and Astheimer P., (2002), "Interactive Toys Environment for Storytelling and Games Applications," In Proc. of StoryTelling Workshop, VSSM'02, Korea.
2002	Kanjo E. and Astheimer P., (2002), "Interactive Environment for Children," In Proc. of the Interaction Design for Children workshop, Endhoven, Netherlands.
2001	Kanjo E. and Astheimer P., (2001), "Hands-On Virtual Worlds," Computer and Fun Meeting, York University, UK.

Table of Content

List of Figures

List of Tables

List of Abbreviations and Symbols

AR	**Augmented Reality**
DOM	**Direct Object Manipulation**
GUI	**Graphical User Interfaces**
HCI	**Human Computer Interaction**
OpenCV	**Intel Open Computer Vision Library**
TUI	**Tangible Interfaces**
HSV	**Hue Saturation Value**
RGB	**Red Green Blue**
ROI	**Region of Interest**
CCD	**Charge Coupled Device**
EM	**Electro Magnetic**
RFID	**Radio Frequency Identification**
CMOS	**Complementary Metal Oxide Semiconductor**
MIT Lab	**Massachusetts Information Technology Lab**
LCD	**Liquid Crystal Display**
CSCW	**Computer-Supported Co-operative Work**
σ	Standard Deviation (STDev)
μ	Mean value
θ	Angle (degree or radian)
τ	Threshold (pixel)
X_w, Y_w, Z_w	Axes of the World frame
X_c, Y_c, Z_c	Axes of the Camera frame
X_f, Y_f, Z_f	Axes of the Focal frame
X_p, Y_p, Z_p	Axes of the Panel frame

1 Introduction

1.1 Overview

Touch is the most critical sense by which we simultaneously perceive compelling information about the surrounding environment and act to alter that environment. Our hands are very skilled devices; they can sense the shape of an object by grasping and by running fingers across its surfaces to build up a mental model. Unlike the passive senses of vision and hearing, touch always precedes perception. Haptic (i.e. touch) experiences activate the human brain not only in the areas that are associated with touch, but also in well-established areas associated with visual processing (Blake et al., 2002).

Object manipulation by hand had been studied for many years by researchers in psychophysics before it began to attract interface designers. These designers have found that the visual representation of an object alone can limit the interaction modalities[1] of the digital interface. This limitation comes from the lack of seamless integration between the two types of entities, i.e., physical objects and computer applications.

More recently, the tactile sensation has been employed as a part of human-computer interfaces. These types of interfaces are concerned with coupling physical objects (i.e. bricks) to and within the digital world. For example, Graspable Interfaces (Fitzmaurice, 1996) and Tangible Bits (Ishii and Ullmer, 1997).

Many of these touch-based interfaces allow people to manipulate a wide variety of physical objects. For example; paper (Mackay & Fayard, 1998; Wellner, 1993), webstickers (Holmquist et al., 1999), bricks (Fitzmaurice et al., 1995), or triangles (Gorbet et al., 1998). Each of these objects does not in themselves represent the information and need the information to be projected or written on their surface. Papers, triangles or even bricks have no meaning without additional information attached to their surface.

[1] Modalities : Perception via visual, auditory, and tactile

However, the complex structure nature of these interfaces restricts the spread of their use as a part of daily life. For example, using extra infrared cameras in the "*EnhancedDesk*" project (Oka et al., 2002), complex video camera setting in the "*Build-IT*" system (Fjeld et al., 1999), electrical-contact sensors and three networked computers in the "*MetaDesk*" project (Ullmer and Ishii, 1997) and high resolution projectors in the "*Interactive Paper*" project (MacKay & Fayard, 1998).

The main contribution of this thesis is a realisation of a novel sensing technology for manipulating multiple physical objects (e.g. cars, buildings or animals) in an entire collaborative environment (i.e. playset or city model) where users do not need to wear special sensing or marked gloves. Neither do the physical objects need to be tagged or marked. This proposed sensing technology is based on the development of novel computer algorithms based on computer vision, which process the frame images grabbed from a low-cost web camera that is mounted in a top-down position to overlook the proposed environment.

These algorithms have a number of features that distinguish the vision system presented in this thesis from existing systems:

- These algorithms are capable of monitoring user actions within the camera's view (e.g. "*moving the toy to a new position*" and "*withdraw hands from the camera view*"). When the user finishes performing the current task by withdrawing their hands from the camera's view, the system automatically responds according to the positions, directions and colours of the moved toys.
- These algorithms also facilitate toy tracking and identification based on new change detection techniques that allow the system to know which toy or toys were moved in the last turn.
- Particular efforts were given to map the positions of the toys in the Interactive Toys Environment to the image coordinate system under various working conditions (e.g. camera vibration or slight background movement).
- Finally, these algorithms can enable different interaction modes which allow the users to manipulate the physical objects according to the application requirements (these modes are turn-based and real-time interaction modes which will be explained in Chapter 4).

In this thesis, the underlying physical objects are referred to as "*Interactive Toys*" and the user environment that results is called the "*Interactive Toys Environment*".

1.2 Why Interactive Toys?

One definition of a toy is that given by Johnson (Johnson, 1984):

"*Toys are a tool of the human child; used to train them in physical skills, help them to develop imagination and to stimulate thinking*"

Most definitions of toys[2] commonly refer to both children and play. Toys, however, can mean different things to different people (e.g. educators, parents, engineers, designers, psychologists, and children). The definition of a toy can also vary by culture, by society, by time, and by the materials and technology available at the time it was made. Toys cater to everyone; children, teenagers, adults, the handicapped and the elderly (Fleming, 1996). Toys can be also considered as general-purpose tools according to the appropriate application.

However, Toys are defined in this thesis as 3-D graspable objects that imitate real-life objects and reflect the information they represents. Users can grasp, move and place them in new positions within the Interactive Environment. Toys' interactivity comes from the responses that the system produces after their movements.

1.3 Characteristics of the Interactive Toys Environment

The Interactive Toys Environment owns the following key characteristics:

Affordances (Norman, 1988): A toy's functionality is perceptually natural and does not have to be taught. For example, novices can quickly learn how to move toys from one place to another. This is unlike traditional computers, input devices such as the mouse where users need to be able to move the mouse in a controlled fashion on a flat surface, and often hold down a button while making these movements.

Multimodality: The Interactive Toys Environment is a multimodal interface due to the use of the tactile, acoustical and visual sensations.

Invisibility: Invisible computing was first predicted by Mark Weiser where computers are so imbedded, so fitting, so natural, that people use them without even thinking about it (Weiser, 1991). The Interactive Toys Environment offers its users an ability to interact

---

[2] The definition of *toy* in the online Dictionary "Hyper Dictionary" includes:
[n] copy that reproduces something in greatly reduced size
[n] a non-functional replica of something else (frequently used as a modifier)
[n] an artefact designed to be played with
[v] manipulate manually or in one's mind or imagination

naturally with computers using ordinary toys. One example will be playing traditional board games on a tabletop while the computer is observing the players movements, and drawing their attention to false moves or giving them some instructions. In this case, the players will not feel that they are interacting with a machine.

***Computer-Supported Co-operative Work (CSCW)*:** In the Interactive Toys Environment, multiple users can move the toys at the same time or in a turn-based manner which cultivates social interaction as well as co-operative thinking, especially for children, who are anxious to have their turn.

***Non-Contact*:** In the Interactive Toys Environment, no specially marked objects are required, nor is a background grid needed. Any object with enough contrast with respect to its background can be chosen. In addition, there is no need for special sensing devices to be attached to a user's hand(s), nor are gloves to be worn.

***Availability for home use*:** The distinguishing feature of the Interactive Toys Environment is that users only need a camera attached to a standard computer to facilitate the interaction. For example, children might incorporate their favourite farm toys on a printed background to interactively play and learn about farm animals.

Reusability: The Interactive Toys Environment is considered as a generic framework with a reusable sensing mechanism which can be used with different types of applications, as will be presented in Chapter 9.

***Direct Object Manipulation (DOM)*:** The Interactive Toys Environment can be viewed as a further evolution of Graphical User Interfaces (GUI) or direct manipulation style of interaction. The essence of a GUI is that the user seems to operate directly on the objects in the computer rather than carrying on a dialogue about them (Shneiderman, 1983).

The Interactive Toys Environment offers users the ability of employing multiple objects to be lifted and moved instead of pointing the cursor and dragging icons on the screen. Thus, this environment retains the essential features of DOM (See Table 1-1).

Table 1-1: *DOM features in GUI and interactive Toys Environment*

DOM features	G U I (Sutcliffe, 1995)	Interactive Toys Environment
Explicit Action	The user points at and manipulates objects on the screen	The user moves toys within the Interactive Toys Environment
Immediate feedback	The results of the user's actions are immediately visible	The results of the user's actions are immediately visible
Incremental effect	The user moves virtual objects continuously instead of sudden jump on the screen	The user moves toys continuously within the environment.
Initiative interaction	Interaction matches the user's conceptual model of how the system should operate	Interaction matches the user's conceptual model of how the system should operate
Learning by onion peeling	The complexity of the system is gradually revealed in layers as the user explores system facilities.	The complexity of the system is gradually revealed in layers as the user explores system facilities.
Reversible actions	All actions can be undone by reversing the sequence of manipulations	All actions can be undone by reversing the sequence of manipulations
Pre-validation	Only valid interactions have an effect, so if the user point at an object and this makes no sense in terms of the current task, nothing happens on the display.	Only valid interactions have an effect, so if the user touches a toy and this makes no sense in terms of the current task, nothing happens till the user moves it to a valid position.

1.4 Structure of the thesis

After this introductory chapter, the remainder of this thesis is structured as follows:

Chapter 2 will present a review of relevant concepts and approaches in the area of Interactive Physical Environments and discusses the shortfall of current formulation.

Chapter 3 briefly presents five sensing technologies and compares their key characteristics with a special focus on computer vision as the chosen positioning and registration technology for the Interactive Toys Environment.

Chapter 4 gives a general overview on the system structure and introduces turn-based and real-time interaction modes.

Chapter 5 presents the panel normalisation algorithm used to perform an invariant and consistent mapping between the coordinate systems relative to the background and the coordinate system relative to the frame image.

Chapter 6 will discuss how to track the toys by detecting changes between frames in sequence with respect to the camera setting and interaction modes.

Chapter 7 will present a robust and fast method for recognising toys of four colours (red, green, blue and yellow) and to detect hand skin colour of different human races and reasonably under various lighting conditions.

Chapter 8 aims at evaluating computer vision algorithms, which were presented in Chapters 4, 5, and 6 in terms of accuracy, robustness and computational time. In addition, it will acknowledge the limitations and give some technical suggestions for enhancement.

Chapter 9 will present several applications developed in this thesis to demonstrate the capacity and stability of the Interactive Toys Environment as an educational but enjoyable space.

Chapter 10 summarises the findings of the current research, discusses the conclusions and makes recommendation for a further work.

The thesis also contains four appendices:

Appendix A provides a brief description of the main software (relevant to computer vision) investigated in this thesis.

Appendix B explains the concept behind pixel connectivity and neighbouring connected components.

Appendix C presents a brief test of web cameras of different manufacturers.

2 Literature Review of Systems and Application area

2.1 Introduction

Since 1990, the discipline Human Computer Interaction saw an explosion in the development of interactive interfaces, where users can interact with information that is physical and reactive to their actions within physical spaces (e.g. board, wall, room or building). The academic literature related to the development of these interfaces and its associated technologies are large and diverse. This chapter provides an overview of these interfaces and in particular the ones with relevance to the Interactive Toys Environment.

The chapter begins with a brief discussion of related sub disciplines of human-computer interaction, including Ubiquitous Computing, Pervasive Computing and Tangible User Interfaces.

The Interactive Surface-Top is then introduced and classified according to several different categories such as space constraints and display techniques. Each of these categories are presented and reviewed in terms of technical structure and design issues. The chapter then turns to an analysis of the different types of physical objects which were used in the related systems and provides a scheme for the classification of these objects.

2.2 Related areas

Research was carried out into bridging the gap between the physical and digital worlds. This research has resulted in the introduction of new forms of computer interfaces, with an activity-centred paradigm aimed at enhancing user interaction through applications that can '*understand*' and '*respond*' to the user's context and activities

Among early efforts at relating the physical and digital worlds was Mark Weiser's work in the field of Ubiquitous Computing (Weiser, 1991) which aims at enhancing computer use by making computers available throughout the physical environment, while making them effectively invisible to the user.

In recent years the terms "*Situated Computing*" (Hirakawa & Hewagamage, 2001) and "*Pervasive Computing*" have also been used to describe a similar concept (Huang et. al. 1999). These developments inhabit a design space that goes beyond the conventional user-interaction provided by the fixed-location desktop metaphor based computing, to more open spaces (i.e. environments).

For example, Druin and Perlin set out to construct an immersive physical environment for adults that responded to movement and touch in real physical spaces (Druin and Perlin, 1994). According to Druin and Perlin, such an immersive physical environment goes beyond desktops to encompass more open spaces such as children's rooms and playing areas.

In another attempt to bring computing power into a user's world, the "*KidsRoom*" (Bobick, 2000) is a computer vision based, reactive and physical play environment for children. Projected images, light and sound effects are all used to transform a space into the setting for a virtual adventure story.

Another example of this type of interface would be the "*LEGO MindStorms Robotic Invention System*" (Ackley & Dooley, 2000). Here, the '*Logo*' programming language has been combined with mechanical turtles, gears, motors, and programmable bricks. With MindStorms, the human-computer interaction is not confined to the desktop and the user space is comparatively open.

However, according to Alborzi and Druin et al. (Alborzi, 2000), there are a number of drawbacks with interactive physical environments, which go beyond the desktop, including:

- They are not easy to modify for different contexts;
- they are not economic;
- they require complicated technology, programs or authoring to support the experience and;
- they were initially confined to research laboratories, largely due to the deficiency of understanding of where the users are and what they are doing in open spaces. For example, while it might be possible to tell that the user is reading a book, it is difficult to recognise what it is the user is reading.

Therefore, many researchers have focused on linking physical objects with digital information within limited spaces such as desktops, board games, children's playsets, tabletops and workspaces.

One development by the Tangible Media Group (in the MIT lab) is the concept of Tangible User Interfaces. One popular paradigm for Tangible User Interfaces is based upon "*Interactive Surfaces*" where physical objects are manipulated by users upon a planar surface. The presence, identity, and configuration of these objects is then electronically tracked, computationally interpreted, and graphically mediated (Ullmer, 2002).

These types of interactive physical environments with limited spaces will be further discussed in the following section.

2.3 Interactive Surface-Tops

Interactive Surface-Tops imply that the locus of the user(s) is around a table or board (either horizontally or vertically oriented) where the physical objects can be positioned and reached. The word "*Top*" was added in here to distinguish these interfaces from other interfaces which use different mediums as interactive surfaces, for example clothes. Interactive Surface-Tops can also be referred to as "*Interactive WhiteBoards*" or "*Interactive Walls*" if the physical objects configured on a near-vertical board, or "*Interactive Workbenches*" if the physical objects are arranged on a near horizontal surface. It should be noted here that the Interactive Toys Environment proposed in this thesis is considered as an Interactive Workbench.

Interactive workbenches can be classified into two major forms, namely, Digitally Enhanced Desktops and Interactive Tabletops. Further, Interactive Board Games and Interactive Children's Playsets are considered as a sub-set of an Interactive Surface-Top. All of these will be explained in more detail in the following sections.

2.3.1 Digitally Enhanced Desktops

Digitally Enhanced Desktops are designed to replace an individual's traditional desk, which depends heavily on the vertical properties of a standard monitor, by integrating activities involving paper-based material, artefacts and digital media.

The pioneering work on physically enhanced desktops was the "*DigitalDesk*" (Wellner, 1993). This has provided a computer augmented environment in which a real physical

desk merges real papers and electronic papers as shown in Figure 2-1. A projector and video cameras are directed onto the desk, and images fed into an image-processing system that can perceive user's action (e.g. writing or drawing) on the desk and respond to them by projecting electronic images onto the desk.

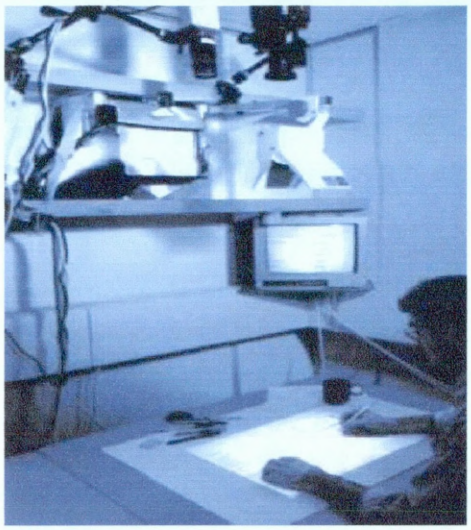


Figure 2-1: *User interacting with the DigitalDesk* (Wellner, 1993)

Another early example of digital desks and a central example of the broader "*Graspable User Interfaces*" approach (Fitzmaurice, 1996) is the "*ActiveDesk*". In the ActiveDesk, two receivers[3] simulate two active bricks that operate on top of the ActiveDesk. Each receiver is a small (25 mm /1 inch) cube that constantly sends positional (x, y, and z) and orientation information to the main workstation. These bricks were used to draw on the ActiveDesk as shown in Figure 2-2. However, the drawing system was limited in that it only provided two bricks for the user to manipulate, and these were connected to the computer with wires which hinder users' interaction.

[3] Two Ascension "Flock of Birds™" 6D input devices.

Figure 2-2: *Graspable Interfaces: Drawing with bricks* (Fitzmaurice, 1996)

The "*metaDESK*" from the MIT lab is also considered as a digital desk with a complex setting that consists of a horizontal back-projected graphical surface; the "*activeLENS*", an arm-mounted LCD screen; the "*passiveLENS*", a passive optically transparent "*lens*", the "*Phicons*", physical icons; instruments which are used on the surface of the desk. The physical objects and instruments are perceived by an array of optical, mechanical, and electro-magnetic field sensors embedded within the metaDESK (Ullmer and Ishii, 1997).

By placing a small Phicon of MIT's Dome onto the desk, a two-dimensional map of MIT appears underneath on the desk, bound to the Dome object at its location on the map as shown in Figure 2-3.

Figure 2-3: *MetaDesk system* (Ullmer and Ishii, 1997)

One of the most recent digital desks is the "*EnhancedDesk*" (Oka et al., 2002) which allows users to use their own hands for direct manipulation of both physical and projected objects. As shown in Figure 2-4, the EnhancedDesk is equipped with an infrared camera, a colour video camera, two LCD projectors, and a plasma display. The infrared camera is used for detection of the user's hands and fingertips which point at which object should be registered and where an object should be looked for.

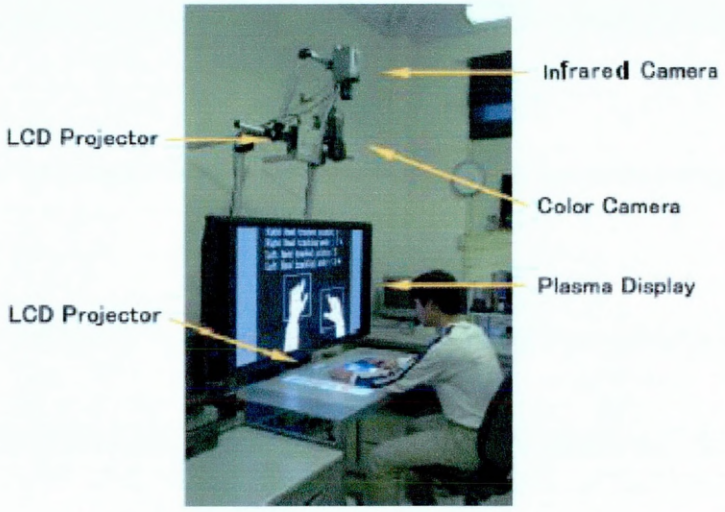


Figure 2-4: *EnhancedDesk system* (Oka et al., 2002)

2.3.2 Interactive WhiteBoards

Interactive Whiteboards, or "*Interactive Walls*", are designed to assist or replace a typical white board, or artist tables. They can be placed vertically on a wall or stand at an angle.

An interactive whiteboard was used in the "*TransBOARD*" system (Ishii, 1997). This combines a digital whiteboard with magnetically backed, business card-sized objects referred to as "*hypercards*". Each hypercard is encoded with a barcode which could be scanned to copy the whiteboard's current contents into the card. These contents could later be accessed over the web (see Figure 2-5).

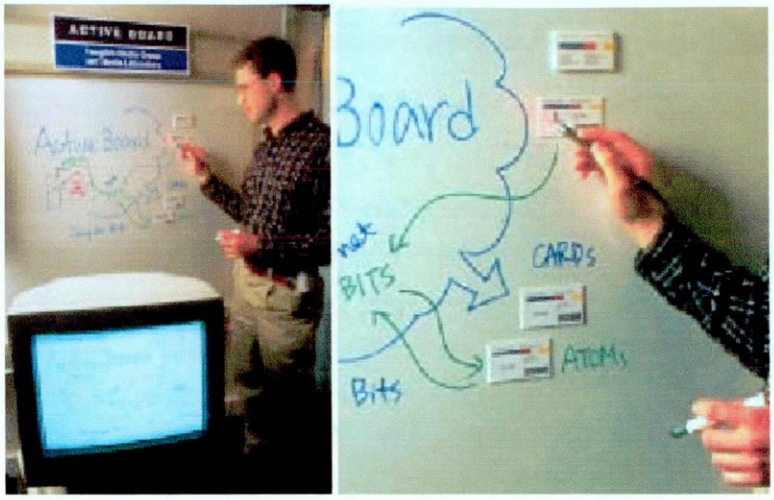


Figure 2-5: *TransBOARD system* (Ishii and Ullmer, 1997)

Another example of the interactive boards is SenseBoard, developed by Jacob et al. (Jacob, 2002) at MIT. The SenseBoard consists of a vertical panel, mounted rather like a portable whiteboard. Small rectangular plastic "*pucks*" (i.e. tags) can be placed on the board and stick there magnetically. Each time the user moves a puck, the board sends the identity and the grid location of each of the pucks to a computer and video projector, which then projects general information onto the board and specific information onto each puck (Jacob et al., 2002) as shown in Figure 2-6.

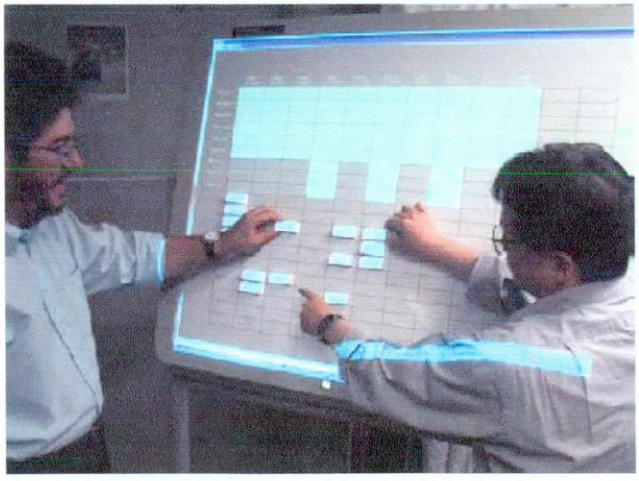


Figure 2-6: *Users interacting with the SenseBoard* (Jacob et al. 2002)

2.3.3 Interactive Board Games

Only a few efforts have been made to date to see how computers and sensing technologies can enhance board games. Mandryk et al. have introduced a hybrid board game-video that is referred to as "*False Prophets*" (Mandryk et al., 2002). The playing surface contains an array of infrared photo-transistors, each of which corresponds to a hexagon in the game. Each character (playing piece) contains an infrared emitting diode. The pieces emit a pulse that is sent though the phototransistors to the serial port and interpreted by the game software. Pieces also have buttons that are pressed to correspond to actions in the game. The display system of the game consists of both the tabletop projection for public information as well as handheld computers for private information. The game map is updated as the player moves around the board.

Another example of interactive board games is the "*Coach*" table (Metoyer & Hodgins, 2000) which uses an infrared vision system to track objects on the table. In this interface the user positions figurines on a miniature electric football field. The figurines have simple handles for pre-specifying their desired direction of motion. When switched on, the field vibrates, causing the figurines (which represent players) to move across the surface (board surface). Offensive and defensive figurines collide and move forward until progress of the ball is stopped and the play is finished (see Figure 2-7).

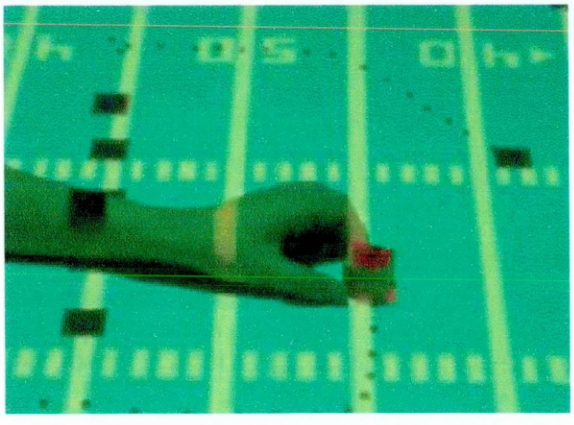


Figure 2-7: *User interacting with the Coach game* (Metoyer & Hodgins, 2000)

2.3.4 Interactive Tabletops

Tables have great potential for collaboration. People naturally gather around tables for discussions as they generally offer a large working surface. Furthermore, artefacts on a table can be examined from various viewpoints. Combining digital technology with tabletops provides the following types of interaction:

- Touching, grasping and moving physical objects such as tools, bricks, toys and pens
- Visualising (by screen display or projection) data in various orientations
- Interpreting hand movements and gestures

Tabletop environments can be classified into two categories according to the display configuration, Display Tabletop and non-Display Tabletop as follows:

Display Tabletops

Many researchers have described some prototypes involving top-projecting a display onto a traditional table. For instance, SenseTable by Patten, Ishii et al. (Patten, 2001) presented a system that electro-magnetically tracks the positions and the orientations of multiple wireless objects (i.e. Pucks) on a tabletop display surface. Users interact with the digital content of the system by touching graphical representations projected onto the desk.

When the puck is touched, the microprocessor inside it detects a capacitance above a certain threshold, and turns that puck on. These Pucks do not represent information, since all have the same appearance and are relatively large to allow digital information to be displayed on their surface-top as shown in Figure 2-8. Moreover, the SenseTable system can only track two pucks at a time.

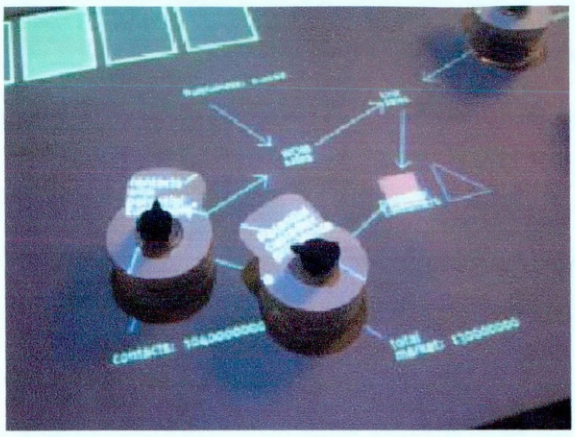


Figure 2-8: *Information is being projected on the active Pucks and the SenseTable* (Patten, 2001)

Another example of display tabletops is the "*DiamondTouch*". This is a multi-user technology for touch-enabled tabletop front-projection displays (Dietz and Leigh, 2001). It works by transmitting signals through antennas in the table which are capacitively coupled through the users and their chairs to receivers, which identify the parts of the table each user is touching. This information can then be used by a computer in the same way as mouse or data tablet. The DiamondTouch system requires a conductive chair and a reasonable electrical isolation between users to enable the system, which is violated if two or more users (or their chairs) are touching (see Figure 2-9).

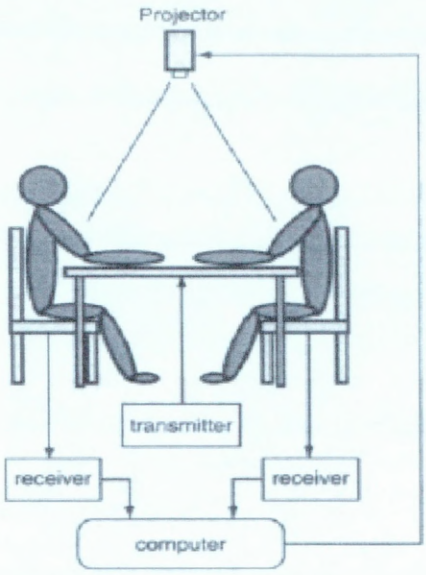


Figure 2-9: *DiamondTouch system* (Dietz and Leigh, 2001)

In contrast, the "*MetaDESK*" (Ullmer and Ishii, 1997) displays the output of the system onto the tabletop using back-projection instead of a front projection display.

Other tabletop systems incorporate a vertical display attached to one side of the table alongside the front or back projection. For example, a prototype system called "*BUILD-IT*" (Rauterberg et al., 1997) uses a video-based interaction and supports engineers, grouped around a table, in designing assembly lines and in building plants. The BUILD-IT system incorporates a vertical display attached to one side of the table and a horizontal working area with several views where the users can select and manipulate small blocks as shown in Figure 2-10.

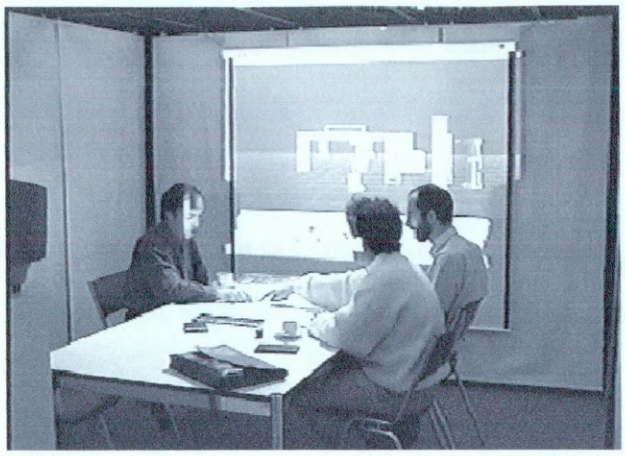


Figure 2-10: *Group of users interacting with BUILD-IT system* (Rauterberg et al., 1997)

Another tabletop system that has been designed to provide vertical displays near the table is "*Tangible Viewpoints*" (Mazalek et al., 2002) which provides a physical interface of "*Pawns*" as shown in Figure 2-11. These Pawns are similar in size and shape to small dolls on a table. Visual story content is displayed on a flat-panel screen and a special projector stand is constructed to project story related graphics on the table.

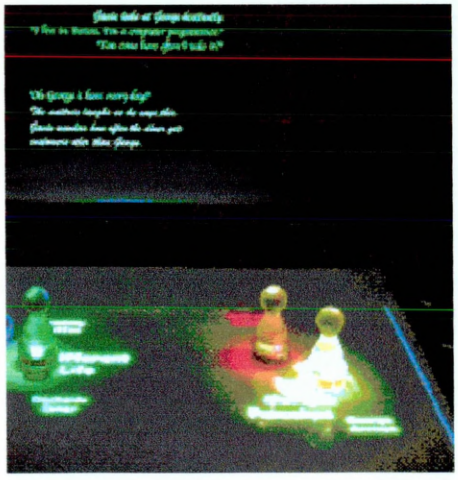


Figure 2-11: *The display setting of the Tangible Viewpoint system* (Mazalek et al., 2002)

An extensive and detailed literature review pertinent to display tabletops can be found in (Scott et al., 2003) who have investigated past and present digital tabletop displays and suggested guidelines for future tabletop displays that provide natural interpersonal interaction, transitions between activities and accessing shared physical/digital objects.

In general, the drawbacks of the display tabletop environments can be summarised as follows:

- Most display tabletops have an inappropriate orientation of the virtual information when users move around the table.
- Currently, only a few tabletop display systems support synchronous collaboration. The majority of these systems require users to move only one active input channel (i.e. physical or virtual object on the table) at a time.
- Projection technology (e.g. front-projection, back-projection and self-illuminating) affects the viewing angle, brightness, and robustness of the interactive systems.
- Projecting the output on the interactive area of the working (or playing) surface can often cause interference between the projected display, shadows and the components of the physical surface such as physical objects and hands.
- Back projection requires considerably more space on the tables.

Non-Display Tabletops

Visual feedback can be displayed on a vertical screen instead of by front or back projection. For example, Sugimoto et al. (Sugitomo, 2000) have designed a non-display tabletop system referred to as "*ePro*". In the ePro system users can construct a town by placing pieces such as houses on a sensor embedded board as shown in Figure 2-12. The radio frequency sensing system recognizes the arrangement of pieces on the board. It then visualizes environmental changes to the town through a vertical screen display. All objects used in the ePro system are tagged while the readers (which read the radio signals) are embedded in the board.

Figure 2-12: *Users interacting with the ePro system* (Sugimoto, 2000)

In the "*RealReality*" project (Ernst, 1999), users can interact with physical models on a tabletop with only an on-wall projection display. However, users need to wear sensing gloves that are referred to as a "*Grasp Tracker*" which might hinder their interaction with the system (see Figure 2-13).

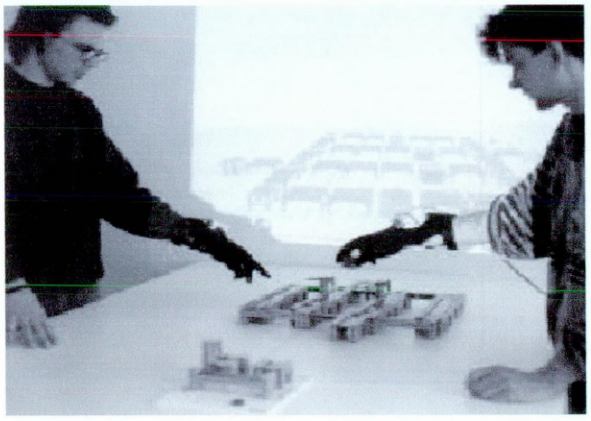


Figure 2-13: *Two users interacting with the RealReality system* (Ernst, 1999)

The "*Triangles*" system (Gorbet et al., 1998) consists of a set of identical flat, plastic triangles, each with a microprocessor inside and magnetic edge connectors and is considered a non-display interface. When the pieces are connected to each other, specific connection information is sent back to a computer that keeps track of the system's configuration. Specific two and three-dimensional configurations of the pieces can trigger application events (see Figure 2-14).

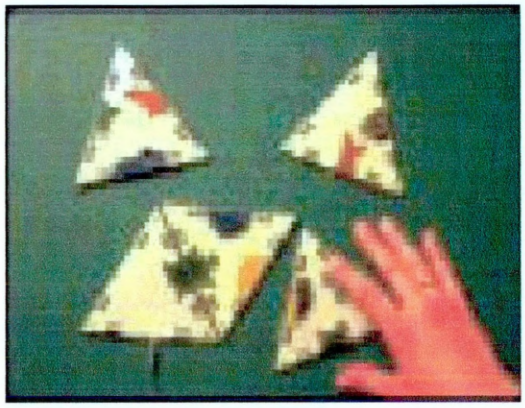


Figure 2-14: *Interaction with triangles to access web contents*

2.3.5 Interactive Children's Playsets

Children influence toy manufacturing and some companies are moving beyond games that simply allow control of the actions on the screen with a joystick or a keyboard. Instead, children are being given the ability to move their hands on playsets and hence to simultaneously control the on-screen action. These types of playsets are referred to as "*CD-Playsets*" and range from simple toolkits, which can be added to an ordinary keyboard like the commercial products of Hasbro Interactive "*Playskool Store*" and "*Tonkia Workshop*", to the advanced "*Zowie Playsets*" (Shwe & Francetic, 2000) that hook up to the computer to partially or completely replace the mouse and keyboard (see Figure 2-15). Zowie Entertainment, now part of the LEGO™ Group, released this breakthrough toy using multiple-object identification technology. In Zowie playsets, toys have special bases which when placed on the playset make them come to life on a computer screen. Although Zowie technology allows fast tracking, the hardware only

provides information regarding the identities and positions of objects in relation to specific locations.

Figure 2-15: Zowie Playset (Shwe & Francetic, 2000)

A similar system was introduced by the Gesture Narrative Group at MIT in the form of "*SAM*" (Cassell et al., 2000). SAM is a physical interface that allows children to engage in natural storytelling play with one figurine and a toy castle, in collaboration with a virtual playmate who shares access to the castle as shown in Figure 2-16.

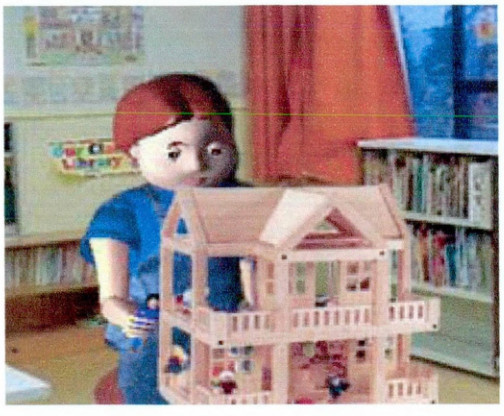


Figure 2-16: *SAM project* (Cassell et al., 2000)

More examples can be given from specialised companies in children products such as the interactive "*Simpson*'s *playsets*" from Playmates[4] (see Figure 2-17) and the "*Talking Grill*" from Hasbro Interactive[5].

Figure 2-17: *Simpson's Playset*

2.4 Discussion of the Related Systems

A brief comparison of nine systems which are the most related to the Interactive Toys Environment is provided in Table 2-1. The aim of these classifications is to provide an overview as to where the Interactive Toys Environment stands in relation to the literature. This comparison is based on six criteria: the complexity of the system, high cost/low cost, tagged/on tagged physical objects, sensory backgrounds/ non-sensory backgrounds, usage of sensing gloves/non sensing gloves.

[4] www.playmatestoys.com
[5] www.hasbro.com

Table 2-1: *Comparison of the Related Systems*

System	Criteria					
	Complex structure	High cost	Ease of use	Tagged or marked physical objects	Girded background	Special sensing gloves
DigitalDesk (Wellner, 1993)	Yes	Yes	No	No	No	No
Build-IT (Rauterberg, 1997)	Yes	Yes	Yes	No	No	Yes
ActiveDesk (Fitzmaurice, 1996)	Yes	Yes	Yes	Yes	Yes	No
metaDESK (Ullmer & Ishii, 1997)	Yes	Yes	No	No	Yes	No
EnhancedDesk (Oka et al., 2002)	Yes	Yes	Yes	No	No	No
SenseTable (Patten, Ishii, 2001)	Yes	Yes	No	Yes	Yes	No
DiamondTouch (Dietz and Leigh, 2001)	Yes	Yes	No	No	Yes	No
Zowie Playsets (Shwe & Francetic, 2000)	No	No	Yes	Yes	Yes	No
SAM (Cassell et al., 2000)	No	Yes	Yes	Yes	No	No
Interactive Toys Environment	No	No	Yes	No	No	No

The conclusion that can be drawn from the table that the complex structure and the cost of the related systems, along with the use of the tags or gloves, have led to introduce the Interactive Toy environment as a natural and convenient interactive system which fit with the objectives of this thesis.

2.5 Properties of physical objects

In this chapter, several systems were discussed which contain physical objects of different characteristics within various interactive physical environments such as interactive tabletops. These physical objects can be classified according to their properties, including:

1- *Usage*

- Generic physical objects which can be applied in many applications. For example, papers in the DigitalDesk.
- Specialised objects that are designed for a specific application. For example, triangles in the Triangles system.

2- *Representation*

The way in which physical and digital objects can be computationally coupled. Holmquist et al. (Holmquist, 1999) suggested a schema for classifying physical objects in term of their link to digital information. This is:

- Tokens - Here, digital information associated with the object is reflected in some way in the physical properties of the token. In another words, tokens do not need further information to be attached or projected on them to understand their functionality. For example, the toys discussed in this thesis.

- Containers - These are generic objects that can be associated with any type of digital information and used to move information between different devices or platforms. For example, the Hypercard in the Transboard system. In addition, papers can be considered as containers as they need drawing or writing on them to represent information.

- Tools - These are attached to digital information on a horizontal display. For example, the Pucks in the SenseTable which need some information to be projected on their surfaces.

3- *Appearance*

For instance, all the Pucks are similar in the SenseTable project. In contrast, many objects have different appearances in the ePro system. However, some physical environments might include similar shaped objects, but with different colours as for example most boardgames.

2.6 Summary

This chapter has briefly described a number of systems that share common ground with the Interactive Toys Environment, which is the centre of interest in this thesis. Some systems share the same tracking capabilities, however, they lack simplicity and robustness; other systems use sensing devices such as gloves or markers which encumber the users. While others require to tag the physical objects to enable object tracking and identification.

This chapter has classified these related systems into different categories starting from the level of the Human Computer Interaction to the display and non-display interactive tabletops.

Similarly, physical objects were categorised on bases of their usage, representations of digital information and their appearance. This type of classification can enable a better understanding of the Interactive Toys as physical objects that represent the digital information on their surfaces.

In the next chapter (Chapter 3), tracking and identification technologies will be reviewed and their advantages and disadvantages presented. Computer vision and its characteristic as a promising tracking technology will be defined and described.

3 Object Tracking and Identification Technologies

3.1 Introduction

In the early stages of the research work reported in this dissertation, it was essential to investigate and review existing sensing technologies, in order to choose a suitable means of tracking and identifying the toys in the Interactive Toys Environment.

As a result of the review, this chapter will provide a general description of the working principles of these technologies, their advantages and their limitations, without delving too deeply into the technical details. A comparison of these technologies through a table of selected characteristics is then presented according to a defined set of metrics.

Computer vision, as chosen tracking technology for the Interactive Toys Environment, is broadly defined in this chapter and its use for Human Computer Interaction reviewed.

3.2 Sensing Technologies

Many researchers such as Michahelles & Schiele and Vildjiounaite (Michahelles, 2003; Vildjiounaite, 2002) have studied and evaluated sensing technologies for physical interaction. This section will briefly present some of these technologies; only technologies with the possibility of being used in Human computer Interaction will be presented.

3.2.1 *Electro-Magnetic tracking systems*

Electro-Magnetic (EM) tracking technology comprises an emitter, located at a fixed position in space, and a sensor, which is attached to the user. Both sensor and emitter consist of three, orthogonal, electromagnetic coils. The emitter produces a magnetic field, which generates a current in the sensor coils as it is moved through the field. The induced currents are then used to determine the user's position.

The largest drawback of magnetic trackers is that they are very sensitive to the presence of metal. This drawback results in mismatches between the tracking sensor's true position and the position as reported by the tracking system. This can reduce the level of accuracy to more than 10 cm, which is not acceptable in many applications. The advantage of EM trackers is that they can be moved freely and are not perturbed by non-metallic objects such as the user's body. In addition, EM tracking systems do not rely on line-of-sight observation.

Current virtual environment systems commonly use electromagnetic position trackers to sense operator head and hand positions to generate virtual environment simulations. The most popular magnetic trackers are manufactured by "Polhemus[6]" and "Ascension[7]".

3.2.2 Acoustic

Acoustic trackers use an ultrasonic source that produces pulses that are received by a set of microphones, usually arranged in a triangle. The time at which a pulse reaches the different microphones gives the source position and orientation. They have the same latency problems as EM trackers and, while not affected by metal, they rely on a direct line-of-sight and are affected by anything that comes between the source and the microphones, such as a body part. They can also suffer from acoustic reflections if surrounded by hard walls, other audio sources producing ultrasound or other acoustically reflective surfaces.

Attempts to bring the sense of hearing to computers interfaces are growing. One of these is the StoryMat project (Ryokai & Cassell, 1999) (see Figure 3-1). This uses ultrasonic triangulating technology embedded in a small stuffed animal to translate its movement on a mat. While a child is squeezing the stuffed animal (i.e. while the mouse button inside the stuffed animal is being pressed) the computer records the child's voice and recognises the two dimensional coordinates of the stuffed animal. When the child releases the stuffed animal, the voice and the movement data are saved as a sound and a text file of coordinates.

[6] www.polhemus.com
[7] www.ascension-tech.com

Figure 3-1: *StoryMat* (Ryokai &Cassell, 1999)

3.2.3 Mechanical

Mechanical trackers connect the physical object to a point of reference. The object position is then measured mechanically from different positions and compared to the initial reference point. Mechanical trackers can be both accurate (0.1 - 2.5 mm) and quick. However, they severely restrict the user's range of motion. The most commonly used mechanical tracker is the 'Boom' by Fake Space Labs[8].

3.2.4 RFID Tags

Radio Frequency Identification (RFID) tagging is a technique used for keeping track of physical objects or people by attaching RFID tags on them. A basic RFID system consists of three components:

-An antenna or coil

-A reader with a decoder

-A Radio Frequency (RF) tag electronically programmed with unique information

The antenna emits RF signals to activate the tag and when an RFID tag passes through the signal zone, it detects the activation signal. The reader then decodes the data written in a memory chip embedded in the tag and passes it (i.e. the data) to the host computer for processing.

[8] www.fakespacelabs.com

RFID tags are categorized as either active or passive. Active RFID tags are powered by an internal battery and they are typically read/write (i.e. tag data can be rewritten or modified).

Passive RFID tags operate without a separate external power source and obtain their operating power from the reader. Passive tags are consequently much lighter than active tags, less expensive, and offer a very long operational lifetime.

The significant advantage of all types of RFID systems is the non-line-of-sight nature of the technology. RFID tags can also be small due to their relative simplicity. For example, RFID tags can be embedded inside security badges, which can allow automatic access to secure doors that are equipped with tag readers.

However, RFID tags have a number of limitations such as limited operating range (2-10 cm approximately) and objects need to be tagged in advanced to enable tracking.

RFID for Interactive Physical Environment

Want et al. (Want, 1999) has proposed the use of RFID tags as tools for linking physical objects with the virtual world in a number of system prototypes such as augmented books linked to the Amazon website.

In this context, the "*ePro*" system has been developed (Sugimoto, 2000). ePro uses a sensing board with RFID tagging technology, which creates an immersive environment by giving users visual and auditory feedback in response to their manipulations on the board. The system board used an antenna arranged like a checkerboard to identify the separate locations of multiple tags attached to small physical pieces on the board.

Figure 3-2 shows how ePro system is used. Users sit around a board and set it up by arranging geographic objects. Each user puts a piece on the board. Every time a user completes his/her move, the computer simulation recognizes the arrangement of pieces on the board, calculates the status of the town, and visualizes it through a LCD projector (or a wide-screen display).

Figure 3-2: *Users interacting with the ePro system* (Sugimoto, 2000)

3.2.5 Optical Tracking

Optical sensors track the positions of one or more actively illuminated markers placed on the tracked object and use geometric triangulation to determine the locations of these markers. Optical trackers in general have high update rates (up to 2500 Hz) and high accuracy (0.1 to 0.5 mm). However, the line of sight should be maintained, as any obstacle between sensor and source degrades performance. Ambient light and infrared radiation also adversely affect performance. As a result, the environment must be carefully designed to eliminate these causes of uncertainty.

3.2.6 Computer Vision

Definition

The purpose of computer vision is to allow the computer to 'see' through a video camera. The camera feeds continuous frames to image processing algorithms that make decisions about real physical objects and scenes in an attempt to imitate the functionality of the human visual and recognition system.

A combination of the processing power of current computer algorithms and the availability of video cameras has led to breakthroughs in computer vision. In structured environments, it is now possible to enable a computer linked to a video camera to detect the presence of a face, compute its gaze and analyse its expression. It is also possible to

track head, arm, hand and finger movements in real-time, as well as to recognise gestures. All these techniques are passive, with no need to hamper the user with spatial gloves or markers. The information required is provided by a video camera or a stereo pair of cameras looking at the user, and is processed by the computer.

Two important considerations in any vision system are the sensitivity and the resolution. Sensitivity is the ability to see in dim light, or to detect weak impulses at wavelengths invisible to the human eye. Resolution is the extent to which a system can differentiate between objects. In general, the better the resolution, the more confined the field of vision.

Computer Vision Applications

Computer vision is used in various industrial and medical applications. Examples include:

- Electronic component analysis
- Signature identification
- Optical Character Recognition (OCR)
- Handwriting recognition
- Object tracking and identification
- Pattern Recognition (PR)
- Materials inspection
- Currency inspection
- Medical image analysis
- Human Computer Interaction (HCI)

Computer Vision System Components

Computer vision systems are comprised of two primary components, a digital camera as an image capture device and the image processing software.

Unlike traditional cameras that use film to capture and store an image, digital cameras use a solid-state device called an "image sensor". These sensors are fingernail-sized silicon chips containing millions of photosensitive diodes which record the intensity of the light that falls on them. The brightness recorded by each diode is then stored as a set of numbers that can be used to set the colour and brightness of dots on the screen.

Charge Coupled Devices (CCDs) have been the main image sensors used in digital cameras having been developed through their use in astronomical telescopes, scanners, and video camcorders. However, there is a new challenger in the form of image sensors that use Complementary Metal Oxide Semiconductor (CMOS[9]) technology.

The most economic form of digital camera is the web camera (or webcam). These were initially built to transfer video over the World Wild Web. Web cameras generally use CCD[9] sensor technology and can be easily installed and used, connecting to the computer using the Universal Serial Bus (USB) port at a frame rate of the order of 15 to 30 fps and with sufficient resolution (e.g. 640 x 480 or better). The popularity and affordability of web camera make them very suitable for computer vision applications, in particular Human Computer Interaction.

Computer Vision for Human Computer Interaction

In the last ten years, an increasing number of researchers in a variety of areas of computer studies have added perceptual capabilities such as speech and vision to computers. In particular, Computer vision and other direct sensing technologies have progressed to the point where it is possible to reliably detect people's activities and respond to them in real-time (Cipolla, 1998).

Several techniques were found in computer vision to enable users to interact with computer interfaces using video cameras. For example, exploiting facial and head gestures (Bregler, 1998), facial expressions (Bascle & Blake, 1998), finger pointing (Cipolla, 1998; Colombo et al., 1998) and full-body gestures (Yacoob, 1998).

A video interface can provide an intuitive and rewarding interactive experience that can enhance many applications (Marks, 2001). Many ideas have been developed to improve user-computer interaction by using live video, for example the BUILD-IT system (Rauterberg et al., 1997) which was explained briefly in Section 2.2.4. BUILD-IT uses vision-based interaction techniques to support a planning tool where a group of engineers engaged in designing assembly lines can sit around a table and interact in a space of virtual and real world objects. The positions of the users are monitored by one or more cameras.

Researchers have also found that video streaming could be used not just in the form of an input device, but also as a non-encumbering sensing technology to create new physical

[9] http://electronics.howstuffworks.com/digital-camera3.htm

interactive experiences in real environments (Bobick, 2000). Thus, Bobick et al. have developed a vision-based play method referred to as "*KidsRoom*" which enables children to engage in a simple fantasy story and interact with computer-controlled cartoon monsters. Three cameras overlooking a specially designed bedroom provide for the computer vision analysis of the scene. One of the cameras is used to track the children and the bed in the space. The overhead position of the camera minimizes the possibility of one user or object occluding another.

With the availability of high-speed processing, reasonable quality and easy-to-install web cameras, and advances in video-processing algorithms, computer vision technology has become capable of enabling certain classes of applications on typical home computers. One of these applications is computer games. Computer games revolve around using what are quite primitive and limiting interfaces such as joysticks, button-activated game consoles, or the computer keyboard itself. Most people would prefer more natural ways of dealing with computers where they can control the game using their body movements and hand gestures.

For example, the commercial product Intel Play Me2Cam includes a web camera (the Me2Cam) with game software. Once the game is started, the child's physical motions, which are captured by the Me2Cam web camera, control all the activities of the software. These motions can be detected by the computer vision algorithms and interpreted by a dedicated program (D'Hooge, 2001).

Another example of using computer vision for computer games is the new Sony Eye Toy which comes with a web camera that is specially manufactured for PlayStation Game Consol. Computer vision algorithms embedded in the system allows people to control the on screen action using their body movements (Marks, 2001).

3.3 Discussion of the Tracking Technologies

The expression of the characteristics of the investigated tracking technologies in exact and unambiguous values are impossible as they vary from one product to another. However, a brief comparison of the six investigated tracking technologies is provided in Table 3-1. Each method has different advantages and disadvantages in respect to the following:

- *Accuracy* - This is a measure of the error in the position reported by the tracker
- *Update Rate* - This is the rate at which position measurements are reported by the tracker to the host computer

- *Contact or Non-Contact* – Relates to the mode of operation
- *Interference Hazard* – Relates to the ability to operate in unstructured environments.
- *Orientation of the tracked object*

A good position tracker for the Interactive Toys Environment should have a high accuracy (~0.5 cm), a high update rate, and minimal interference hazard without sensor contact to users or objects. Ideally, it should not need any specialized gloves, tagged objects or even a background grid for operation. In addition, it should be easy to use and setup.

The comparison and the descriptions in this chapter show that none of the current technologies is a panacea for the position tracking problem. Each technology has its own advantages that make it suitable for certain applications. However, each technology also has some major drawbacks, for example, distortion and short range in magnetic trackers, low update rates and occlusion in acoustic trackers, occlusion and need for elaborately designed environments in optical trackers, and user discomfort in mechanical trackers.

Hence, computer vision is considered as the most suitable technology for the proposed tracking problem in this thesis. Its accuracy and update rate are acceptable as it differs according to the imaging device and the tracking algorithms. It is also considered as non-intrusive technology, which can accommodate multiple users simultaneously.

Table 3-1: *Objects tracking technologies*

	Criteria				
Type	**Accuracy**	**Bandwidth**	**Interference Hazard**	**Contact/ Non Contact**	**Orientation detection**
Mechanical	0.1-2.5 mm	> 3000 Hz	Physical occlusion	Direct Contact	Yes
Optical	0.1-0.5 mm	100 -2500 Hz	Occlusion	Markers attached to objects	Yes
Magnetic	~5 mm	20 - 100 Hz	Metal Objects	Detectors attached to users	Yes
Acoustic	~1 mm	500-1000 Hz	Acoustic Sources	None	-
RFID tags	3-10 cm	100 kHz -5.9 GHz	-	Tags are connected to objects	-
Computer Vision	vary	fps depends on the camera	Physical occlusion	none	Yes

3.4 Summary

This chapter has presented a number of tracking technologies, which are widely used for positioning and identification purposes. In particular, it has focussed on those that have potential applications in Human-Computer-Interaction.

Computer vision which relies on the imaging device, image processing and programming techniques has been accelerated by the recent integration of fast computers and the availability of simply, low cost web cameras.

By choosing computer vision to facilitate toy tracking and identification, it is possible to develop the Interactive Toys Environment both with sensor-free toys and gloves-free hand interaction.

4 System Overview and Interaction Modes

Introduction

The objective of this chapter is to provide a general system overview of the Interactive Toys Environment in relation to camera configuration and components. In addition, the chapter introduces two interaction modes which are dependent on the ways that users manipulate the toys in the final application. These are the turn-based interaction mode and the real-time interaction mode. The building blocks of these modes will initially be presented in a general manner, with an extensive explanation of each block then being given in chapters 5, 6 and 7.

System Overview

The Interactive Toys Environment consists of a web camera positioned to overlook a number of toys on an interactive area such as a game board or printed-paper to act as background. This interactive area will be referred to as the 'interactive panel'. Input images captured by the camera are passed to object registration and recognition algorithms. Then, the information extracted from these algorithms is sent to the final application to deliver an appropriate audio and visual feedback to the users. In general, both the interactive panel and the toys can be chosen according to the application. However, some restrictions need to be applied in terms of colours, shapes and sizes. These restrictions will be identified when presenting the system algorithms.

Figure 4-1 provides an overview of the Interactive Toys Environment system, where a user's hand is moving a toy on a multicoloured interactive panel and computer screen is displaying visual output, alongside computer speakers to communicate with the user. A toy farm was used as an example application.

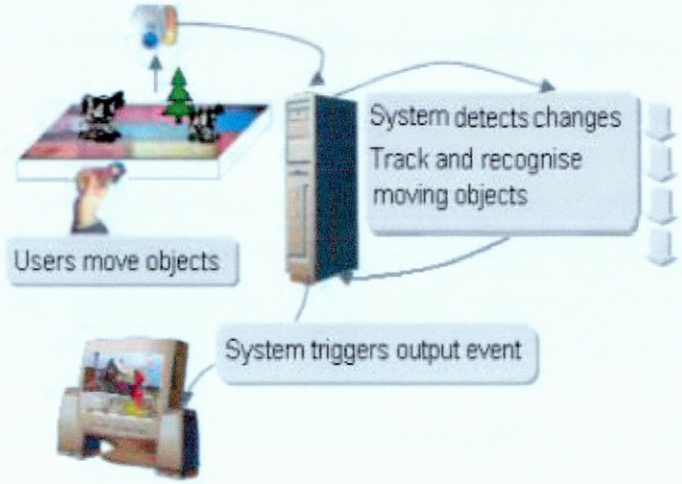


Figure 4-1: *System level overview of the Interactive Toys Environment*

4.3 Interaction Modes

One of the challenging tasks in designing an interactive system is that of finding optimal methods to enable users to deal with the digital and physical information. Therefore, two different methods of interaction can be introduced according to the way that users move toys to control the proposed application. The primary mode of interaction with the system is the turn-based interaction mode with the other mode being the real-time interaction mode.

In the following sections, the design issues associated with these forms of interaction will be discussed:

4.3.1 Turn-based Interaction Mode

In the turn-based interaction mode developed in this thesis, users perform a task by moving a toy or toys inside the borders of the interactive panel and then withdraw their hands from the camera view. The system translates '*hand intrusion*' to process command "*user is performing a task*". Similarly, the system translates '*withdrawing hands*' to a process command "*users have made an input to the system*".

Figure 4-2 shows two frame images taken while a user is playing a game using the turn-based interaction mode. Figure 4-2(a) shows the user's hand is holding a red toy in order to place it on an interactive panel representing a small village as background. Figure 4-

2(b) does not contain the user's hand as he has finished performing the task by placing the red toy on the interactive panel.

(a) *User is performing a task*

(b) *User has made an input to the system*

Figure 4-2: *User's hand interacting with the system in the turn-based mode*

The turn-based mode starts with the user taking a reference image of the background without any toys. This image will be subject to geometrical corrections, which will wrap the interactive panel points to the frame image points. Image warping requires prior detection of the four corners of the interactive panel within the camera's view. If the interactive panel is not entirely visible to the camera, then the system will deliver a warning message to the user to correct the position of the panel. Interactive panel detection and image warping will be referred to as '*Panel Normalisation*'. Panel

normalisation is performed every frame image to enable a turn-based algorithm that is robust and insensitive to small changes in the position of the interactive panel (Panel Normalisation will be discussed in details in Chapter 5).

After capturing the background image and correcting it, the system prompts the user to start the interaction by placing a toy or toys within the area of the interactive panel. While the users are moving their hands in the camera's view, the system will wait. As soon as the user withdraws their hand from the camera view, the system begins to search for moved toys using two frames representing the start conditions prior to user intervention (f_1) and the conditions at the end of that intervention (f_2) with the background as reference (the two frames are also subject to panel normalisation). This process is referred to as '*Change Detection*', which will be described in Section 6.2. Change detection will lead to the generation of segmented regions which represent the moved toys. In the case that more than one toy is moved, then another process referred to as '*Region Labelling*' is required to identify regions. Region Labelling will be explained in details in Section 6.2.6.

After identifying the segmented regions and toy centroid positions, orientations and sizes can be calculated using '*Image Moment Calculations*' which will be discussed in Section 6.3.

Eventually, the colour recognition process is used to identify the toys (Colour Recognition will be presented in Section 7.6. In order to limit the complexity of the object recognition problem, toys of uniform colour are selected to support colour recognition. Lastly, the current frame image f_2 will be stored as f_1 to enable another frame to be fed in f_2.

Figure 4-3 shows a user's hand moving a toy in turn-based mode in order to move the corresponding character in the Baldure's game running on the computer seen in the figure.

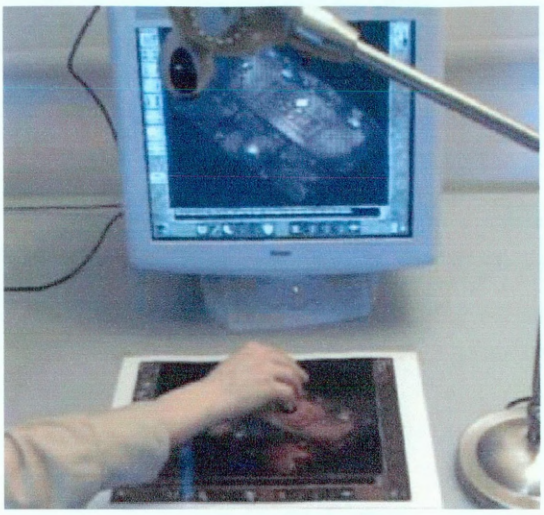


Figure 4-3: *User is being playing with Baldur's game in the turn-based mode*

Figure 4-4 shows the building blocks of the turn-based interaction mode.

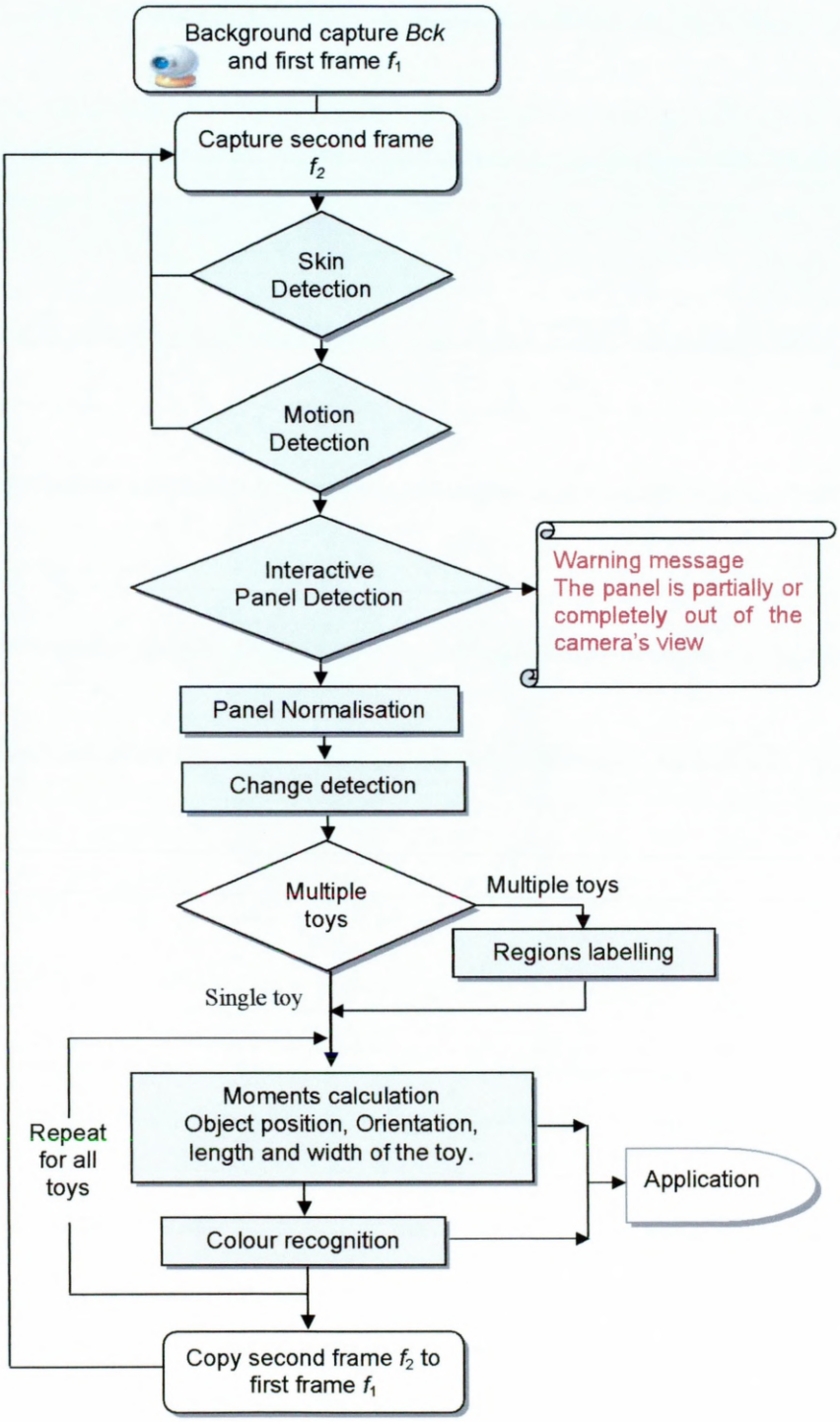


Figure 4-4:*Diagram of the turn-based interaction mode*

How does the system perceive whether the user' hand is currently manipulating toys or not?

Different techniques were found in the literature to identify hands' regions for natural human computer interaction. For example in the Enhanced Desk (Sato, 2000) an infrared camera has been used to segment the hand regions on the desk by setting the temperature range to approximately human body temperature, image regions corresponding to human skin then appear particularly bright in input images from the infrared camera (Figure 4-5).

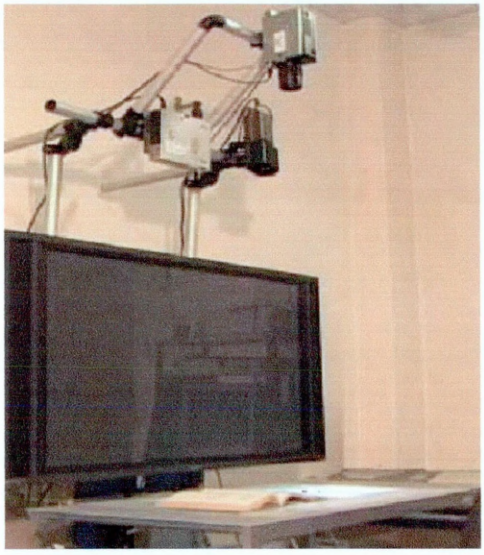


Figure 4-5: *The Enhanced Desk system* (Sato, 2000)

Maggioni (Maggioni, 1993) has presented the use of a specially marked glove for hand segmentation. By identifying the glove with a single camera, the system can estimate hand position and orientation.

Unlike these systems, the Interactive Toys Environment does not need extra equipment such as infrared cameras or marked gloves to identify the hand region. Instead, a novel approach which relies on the combination of skin detection and motion detection techniques was developed to perceive whether the user is moving the toys or not. These techniques will be explained in Sections 7.5 and Section 6.2.5 respectively.

4.3.2 Real-Time Mode

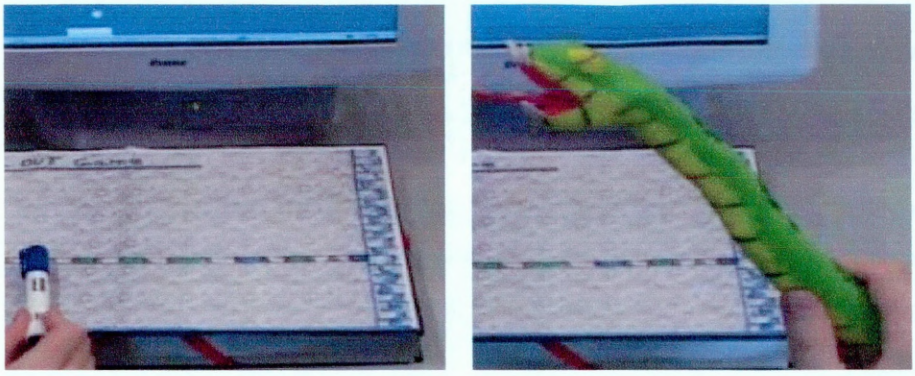
It was shown in relation to the turn-based mode that users have to withdraw their hands after each turn to inform the computer of their input. Another interaction mode was therefore developed, which does not require that users withdraw their hands from the camera's view after each turn. Instead, a toy or toys can be moved continuously and its position are transferred frame by frame to the appropriate application. This type of interaction has been used by a number of researchers for mouse and joystick replacement. For example, Bradski et al. from Intel Corporation have used a small toy aircraft in simple colours to navigate into a virtual scene as a replacement to the ordinary joystick (Bradski, 2001) as shown in Figure 4-6:

(a) Screenshot of virtual space on the computer screen

(b) User moving the physical toy plane as a joystick in real-time

Figure 4-6: *User navigating into a virtual space using Toy Joystick.*

Unlike the Toy Joystick, the real-time mode discussed in this thesis enables a free range of toys to be used. Users might use their pen, or toy snake as shown in Figure 4-7.



(a) An ordinary pen (b) A snake toy

Figure 4-7: *Examples of real-time tracking using different objects*,

Furthermore, the Interactive Toys Environments supports two types of real-time interaction modes in relation to the background. These are:

- Real-time background dependent mode
- Real-time background independent mode

4.3.2.1 Real-time background-dependent mode

In this mode, the user needs to capture the background as a reference image *Bck* at the beginning of the interaction process. Then for every frame image f_2, skin pixels are removed and the '*change detection*' process will segment the image to create a region (or regions) that represent the moving toy over two frame images in sequence (Change detection for real-time background-dependent mode will be described in details in Section 6.3). In the case more than one toy is moved, then '*Region Labelling*' is required to identify each region. After identifying the segmented regions, toy centroid positions, orientations and sizes can be calculated using the 'Image Moments Calculations' (Moment will be described in details in Section 6.3). Colour recognition can also be used to identify the toy's colour. Lastly, the current frame image f_2 will be stored in f_1 to enable another frame to be fed in f_2.

The diagram of Figure 4-8 illustrates the system building blocks for this mode.

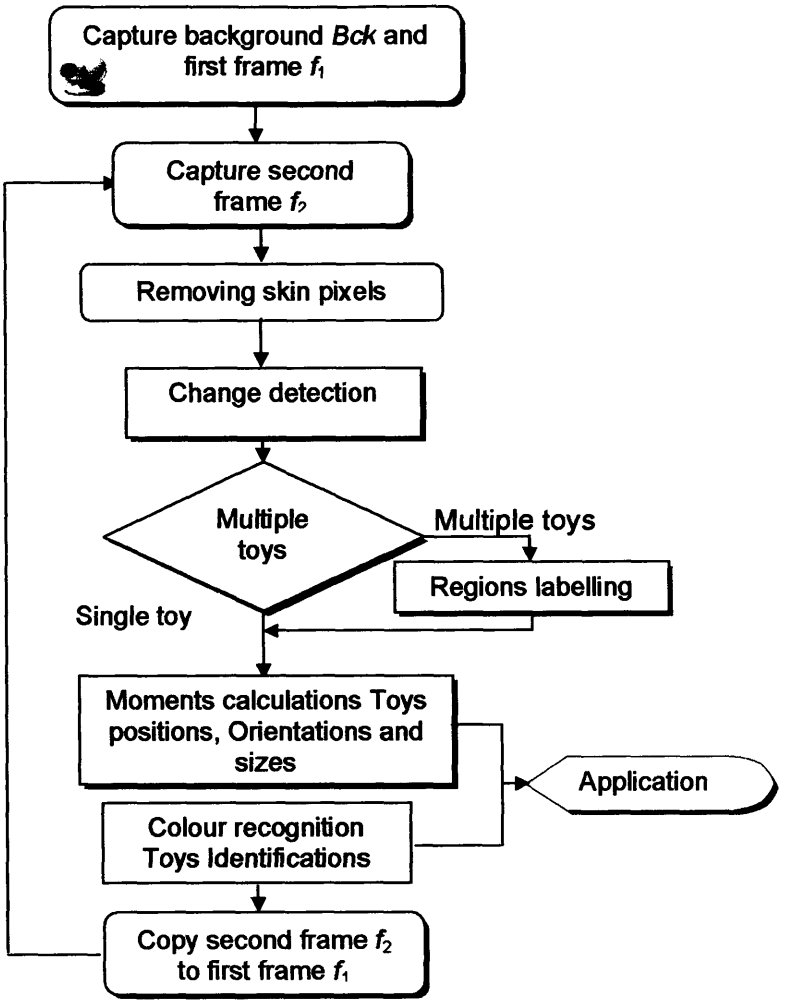


Figure 4-8: *Diagram of real-time-background-dependent interaction mode*

Figure 4-9 shows user's hand moving a plane toy with long handle in the real-time background dependent mode in order to control the virtual character in the "*Chicken Hunt*" game (This application will be described in Section 9.2.2).

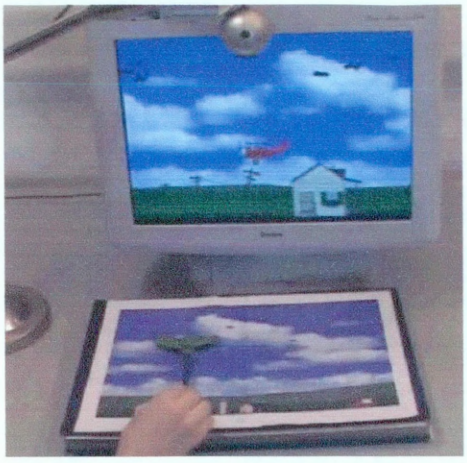


Figure 4-9: *User interacting with Chicken Hunt game in real-time background dependent mode*

4.3.2.2 Real-time background-independent mode

This is the simplest and fastest interaction mode and does not need the background as a reference image. It works by tracking the changes over each of two successive frames and, as a result, user interaction is not affected by any movement of the camera or the interactive panel as the system recovers itself in the next frame.

Figure 4-10 shows the main building blocks of the real-time-background-independent interaction mode with change detection, movement calculations and colour recognition as the main elements.

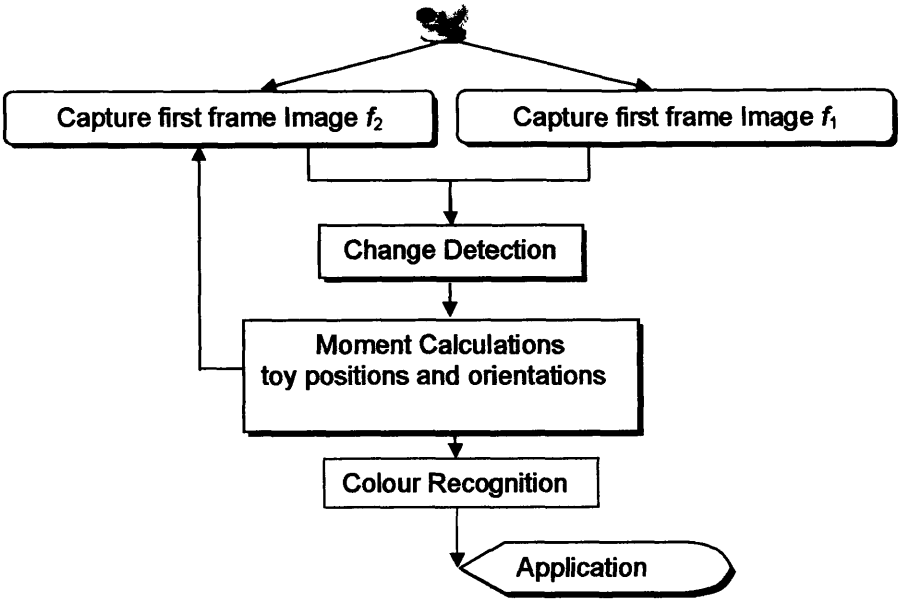


Figure 4-10: *System diagram for the real-time background-independent interaction mode*

Figure 4-11 illustrate an example of employing the real-time interaction mode to control a virtual pedal in a real-time in the "*Breakout Game*" (This application will be described in Section 9.2.3). It is possible to see, the pen position in the camera's view matches the pedal position in the computer screen (The two red arrows show the movement directions of both the pedal and the pen).

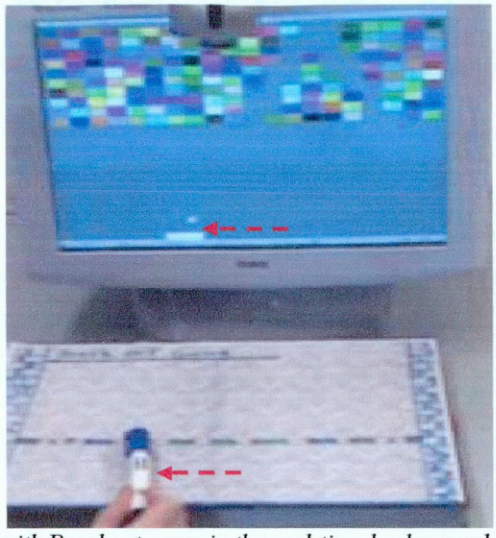


Figure 4-11: *User interacting with Breakout game in the real-time background independent mode*

4.4 Summary

This chapter has presented a system overview of the Interactive Toys Environment based on computer vision algorithms. Camera configuration in relation to the interactive panel was discussed and various interaction modes were introduced. A turn-based interaction mode was described as a natural and effective method to exchange turns between the computer and its users. In the turn-based mode, hand skin colour and rapid motion were checked over the frame images to decide whether the user or users are performing a task or waiting for a response from the system.

In contrast, the real-time interaction mode enables users to engage in active interaction with the computer while their toy's movement is tracked on a frame-by-frame basis.

These interaction modes can be chosen in relation to the application. For example, the real-time interaction mode can be used to control a virtual plane to follow some on-screen birds, while the turn-based interaction mode can be for board games or children's playsets, applications that will be presented in detail in Chapter 9.

The building blocks of the diagrams presented in this chapter will now be explained in detail in chapters 5, 6 and 7.

5 Panel Normalization

5.1 Introduction

It was explained in Section 4-2 how in the Interactive Toys Environment the camera is positioned to view the whole of the interactive panel. It was also shown that a reference image of the background is required in order to detect changes and hence to track the toys. Unfortunately, there are a range of problems and questions associated with the camera configuration. For example, the camera might be tilted or moved by a range of different factors such as air conditioners or people's movements. These problems make it harder to track the moving toys in relation to the interactive panel.

To solve this problem a geometrical solution referred to as '*Panel Normalisation*' was developed and which is presented in this thesis. This chapter will present panel normalisation as a robust technique to enable the computer to track the toys in relation to the interactive panel, even if the camera is tilted or vibrated.

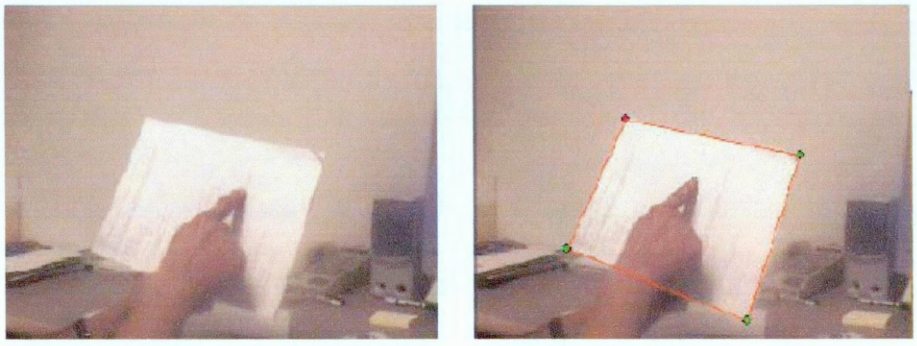
The chapter starts by giving a general background of the problem supported by some examples from the literature. Subsequent sections then present the technical details of the panel normalisation algorithm.

5.1.1 Background

The idea of mapping the corners of the rectangle-shaped panel points to the image plane points has been used by a number of computer vision systems intended for Human Computer Interaction. For example, the DigitalDesk (Wellner, 1993) introduced a self-calibration method to support interaction on the office desk. The system projects four-reference points (thick plus signs) onto the desk, these points can then be extracted by the image processing system, which then performs the mapping between the image and interactive area on the desk.

Researchers at Microsoft's Research Lab have developed a system that automatically detects a quadrangle in an image. In the Visual Panel system (Zhang, 2003), four straight sidelines of the panel can be detected using edge detection and Hough Transform (Hough, 1962). Once the sidelines are determined, the corner points are computed from the

intersection of these lines. The four corners are then used to relate the panel to the original image as illustrated in Figure 5-1.



(a) Input image (b) Tracking result - a tracked panel

Figure 5-1: *The tracking process in the Visual Panel system* (Zhang, 2003)

5.2 Panel Normalisation

The approach taken in this dissertation is directed at choosing an interactive panel of rectangle shape with enough contrast to separate it from its background. This will enable the four strong edges of the interactive panel's to be identified along with their intersection at the four corners. By detecting these corners, an image mapping can be used to map the interactive panel points to the image plane, which is also of rectangular shape.

Panel normalisation in this context requires an interactive panel with a clear border to allow accurate edge detection. Figure 5-2 shows three examples of interactive panels with clear borders, enough to ensure contrast with its background.

(a) Dark interactive panel with lighter uniform background

(b) Dark interactive panel with a green uniform background

(c) Light interactive panel with a clear marked border on a light pink background

Figure 5-2: *Interactive panel types*

5.2.1 Geometrical framework

To simplify the panel normalisation problem a number of assumptions have to be made. Firstly, the frame image is positioned reasonably parallel to the workspace (as the camera looks vertically down). Secondly, the pixels are squares and the principle point, where the optic axis intersects with the image, is at the image centre.

These assumptions are illustrated in Figure 5-3 where an image is taken by a camera looking perpendicularly down on a planar workspace and performing a perspective projection. In addition, the figure shows five reference frames, which are essential for the analysis of the 3-D scene. These frames are:

- Pixel coordinate frame I
- Interactive panel coordinate frame P
- Camera coordinate frame C
- Real image coordinate frame F
- World coordinate frame W

The principal point C and the centre point of the image are both located on the Z_c axis. The four corners (a, b, c, d) of the interactive panel can also be seen on the figure.

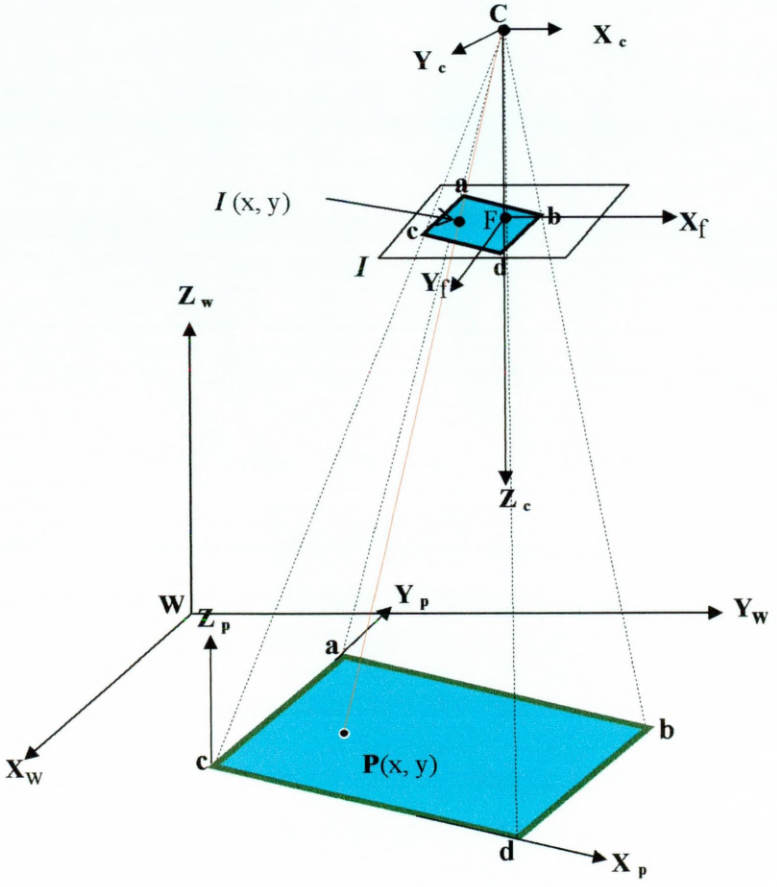


Figure 5-3: *Perspective camera model of the Interactive Toys Environment*

All the points of the interactive panel have a $Z_w = 0$ relative to the world coordinate system W. Thus, the problem can be simplified into a 2-D transformation which maps between the pixel frame (I) and the interactive panel frame (P) as shown in Figure 5-4.

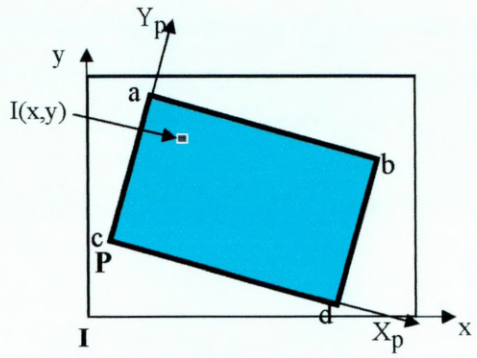


Figure 5-4: *Interactive panel frame P within the Image frame I*

5.2.2 Panel Normalisation Algorithm

The diagram of Figure 5-5 shows different steps of the panel normalisation algorithm developed in this thesis. These building blocks will be explained in detail in Sections 5.2.2.1 and 5.2.2.2:

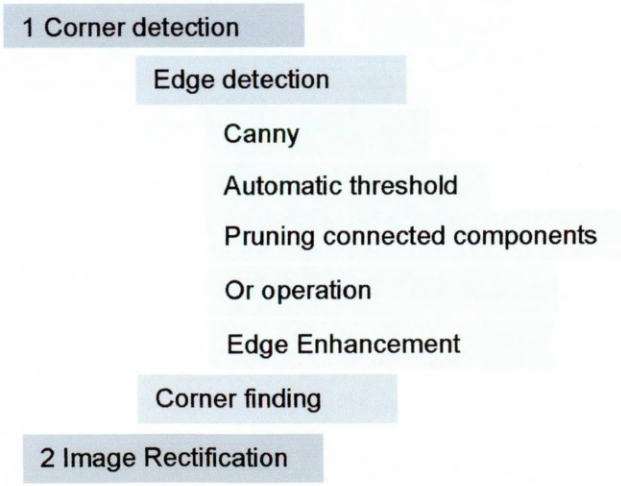


Figure 5-5: *The building blocks of the panel normalisation algorithm*

5.2.2.1 Corner Detection

Corners in digital images are places where two strong edges meet. In the last twenty years, a number of corner detectors have been reported. For examples, consider those by Harris & Stephens (Harris, 1988), Wang & Brady (Wang, 1992) and the SUSAN algorithm (Smith, 1997). Most of these popular corner detectors work on the grey-scale image directly. Thus, they do not need to extract edge points before finding corners, they usually look for any corners in the image without prior knowledge.

In this thesis, a corner detection algorithm was developed to find the four corners of the interactive panel. The algorithm therefore aims firstly at edge detection and then at finding the four corners. The Corner detection algorithm for panel normalisation is made up of two steps, edge detection and corner finding.

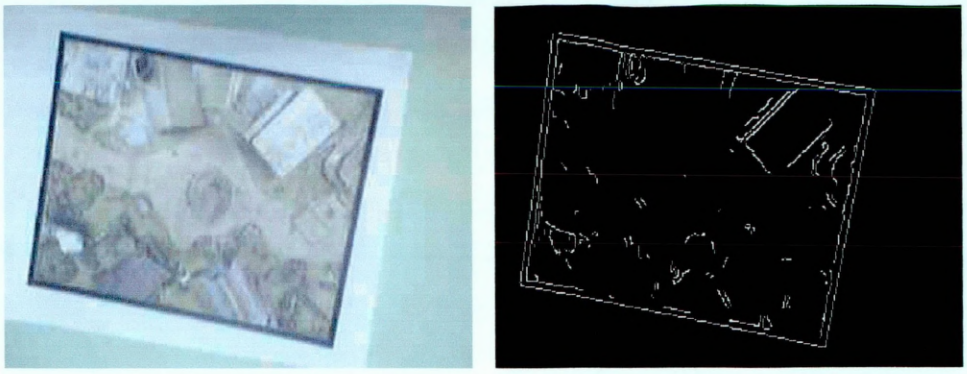
Edge Detection

The aim of this step is to localise the borders of the interactive panel in the image field using edge detection as the first step in the boundary detection process. In this process, potential edge points are taken as corresponding to places in an image where rapid changes in brightness occur (Umbaugh, 1999) and are marked.

Edge detection for panel normalisation consists of five steps as follows:

1-Canny Edge Detection - One of the most popular edge detectors is that developed by Canny (Canny, 1986) who proposed a method for edge detection that produces three classes of edges using two different thresholds. All edges E_h above the high threshold level τ_h are retained and all edges E_l below the low threshold τ_l are rejected. The remaining edges (E_m) are retained only if they are adjacent to the edges E_h or connected to E_h via E_m. The Canny method is therefore less likely to be "fooled" by noise, and more likely to detect true weak edges (Mathswork, 2004).

The best values of the thresholds need to be calculated dynamically based on the image content, possibly fluctuating camera levels and viewing conditions (Rosin, 1997). Threshold values are selected using trial and error testing. Figure 5-6 shows the results of applying the Canny edge detector to an input image of a rectangle-shape panel with a black border.



(a) Original image (b) Resulting image

Figure 5-6: *The application of Canny edge detection*

It is clear from the figure that the Canny edge detector alone is insufficient for corner detection as it makes the straight lines into curves and misses some of the edge pixels, see the missing pixels in the internal edge of the black borders in Figure 5-6(b) which should have been recovered.

2-Automatic Thresholding - If the pixel intensity of the original image *I* lies below the difference between the mean and the standard deviation, then this pixel is set to 1 otherwise it is set to 0, as in:

$$I_{th}(x,y) = 1 \quad If \quad I(x,y) <= (\mu - \sigma)$$
$$I_{th}(x,y) = 0 \quad Otherwise \qquad 5.4$$

The result of the threshold is illustrated in Figure 5-7 for the same input image in Figure 5-6, where thick edge represents the black border of the panel with some noise.

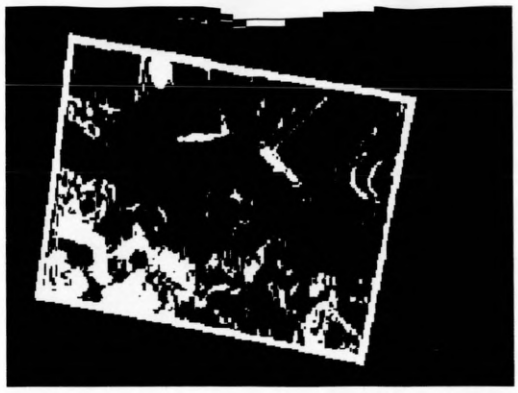


Figure 5-7: *The result of automatic thresholding*

3-Pruning of Connected Components - Pruning connected components[10] is applied on the foreground pixels (pixels of non-zero values). Pixel I_{th}(i, j) is set to zero if eight 8-neighbours are found, while preserved pixels with less than eight 8-neighbours (edge pixels always have less than eight 8-neighbours). The procedure is iterative and continues until every foreground pixel has less than eight 8-neighbours. The pruning process is formulated as:

$$\begin{aligned} I_N(i,j) &= 0 \quad If \quad N_8(I_{th}(i,j)) = 8 \\ I_N(i,j) &= 1 \quad Otherwise \end{aligned} \qquad 5.5$$

Figure 5-8 illustrates an example of a 6×5 binary image of potential edge pixels E and none edge pixels N:

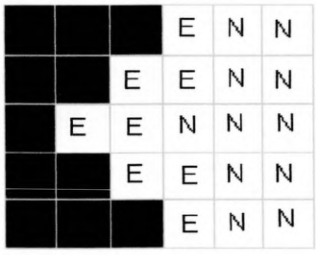


Figure 5-8: *The application of pruning connected components for edge detection*

[10] Connected components background is given in Appendix B.

Figure 5-9 shows the results of pruning connected components applied on the threshold image I_{th}.

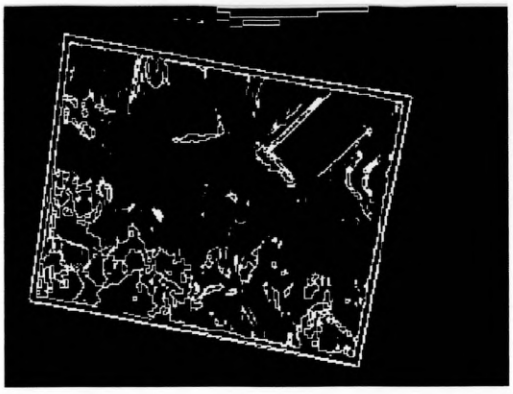


Figure 5-9: *The result image of pruning of connected components*

4-OR operation - It is then possible to get the final edge image by applying the OR operation to the image I_{Canny} and I_{N} as follows:

$$E = I_{Canny} \oplus I_N \qquad 5.6$$

Figure 5-10 shows the result of the OR operation as the edges gets clearer. The edge in the figure is the actual result of combining the Canny edge detector, automatic thresholding and pruning connected components.

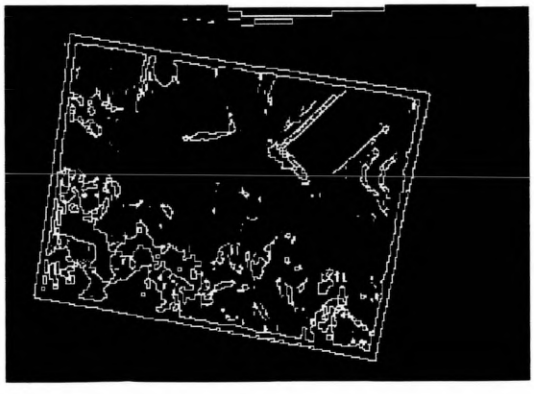


Figure 5-10: *The result of combining Canny, automatic thresholding and pruning using the OR operation*

5-Edge Enhancement and Noise Removal

Noise removal of connected components is a common step for removing undesirable artefacts (thin regions) created during the preceding image processing steps (Ritter & Wilson, 2001).

Because of its simplicity, the region-growing algorithm is selected here for noise removal. Region growing is a procedure that groups pixels or sub-regions into larger regions based on predefined criteria (Zucker, 1976). It is used here to remove small regions and retain the bigger ones using the following criteria:

- Count pixels in each region of each 8-neighbours connected component.
- If the number of pixels is lower than a predefined threshold T_h, then this region is considered as noise and its pixels will be set to zero. Threshold T_h is selected using trial and error testing.
- Figure 5-11 shows the results of applying noise removal using region growing on the edge image E from figure 5-10. The internal and the external edges of the interactive panel's border are clearly outlined.

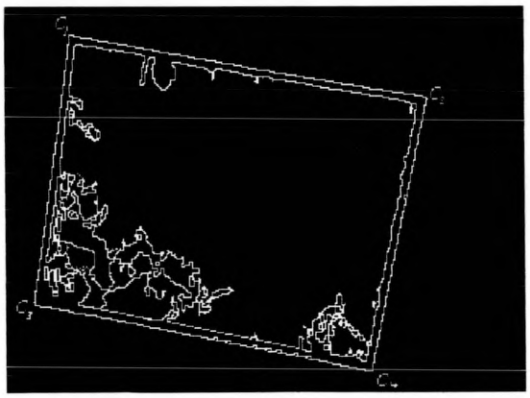


Figure 5-11: *The result of edge enhancement and noise removal*

Finally, Figure 5-12 shows the five steps used to get clearly edges from the interactive panel in preparation for corner finding.

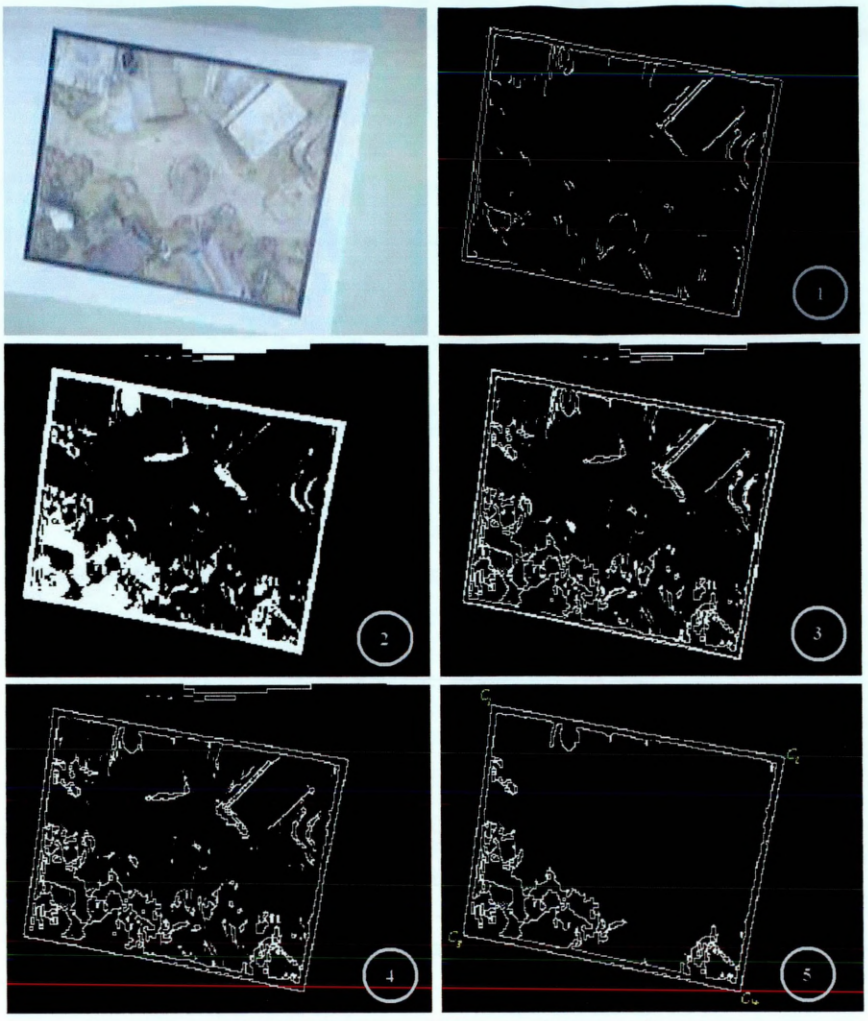


Figure 5-12: *The five stages of the edge detection algorithm*

Corner Finding

To find the four corners of the interactive panel, a simple algorithm was developed which takes the edge image as an input, finds the minimum and the maximum of the x and y coordinates of the edge pixels and returns corner coordinates. The algorithm is structured as set out below in which:

Algorithm: Finding Corner

E is the binary edge image.

$C_k(x, y) \mid k \in [0, 3]$ are the four corners of the interactive panel.

w is the width of image *E*

l is the height of image *E*

X is a temporary one-dimensional array

Y is a temporary one-dimensional array

Step 1:

```
n = m=0; // initialize counters
    for i=1 to w
        for j=1 to l
            if (E [ i , j] ≠0) {
            X [n] = i; // Store all x coordinates of the Edge pixels in one row array X
            Y [m] =j; // Store all y coordinates of the Edge pixels in one row array Y
                N = n+1;
                M = m+1;   }
        }
}
```

Step 2:

```
C1x = minimum (X); // find the minimum x coordinate of the edge pixels
C2x = maximum (X); // find the maximum x coordinate of the edge pixels
C3y = minimum (Y); // find the minimum y coordinate of the edge pixels
C4y = maximum (Y); // find the maximum y coordinate of the edge pixels
```

Step 3:

```
for i = 1 to w
    for j=1 to l
        if (E( i , j ) ≠0 )
            {   if (i = C1x ) then C1y = j;
                if (i = C2x ) then C2y = j;
                if (j = C3y ) then C3x = i ;
                if (j = C4y ) then C4x = i ;
            }
    }
}
```

5.2.2.2 *Image rectification*

With the four corners of the interactive panel established, a simple image warping (i.e. mapping of the corners to the actual corners of the image plane) is performed for each image in sequence. The result is a rectified image of the interactive panel that fills the image frame.

There are different approaches for image warping such as perspective, affine and bilinear (Glasbey, 1998). This section will show that warping a rectangular panel shape to the image frame is possible using two approaches, namely:

1. Affine warping- rotation, scaling and translation
2. Bilinear warping

Affine Warping - Rotation, Scaling and Translation

This method uses two corners of the interactive panel (e.g. 'a' and 'b' in Figure. 5-13), assuming it has a rectangle shape since the image frame is parallel with the workspace.

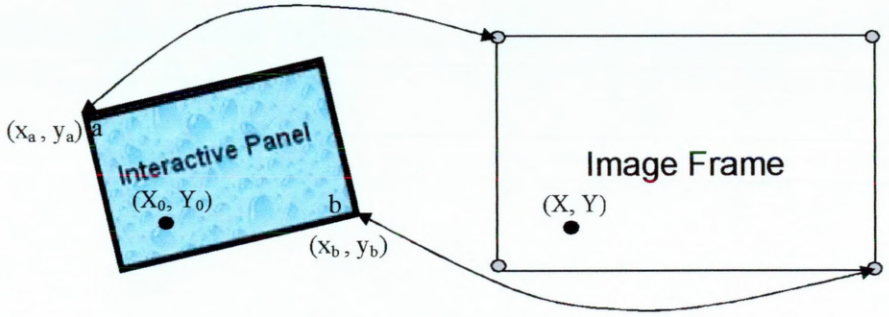


Figure 5-13: *Affine warping*

The transformation can be done by composing a rotation R, a scaling S and a translation d_X, d_Y as given in Equations 5.7, 5.8 and 5.9 respectively.

$$\begin{bmatrix} x_o \\ y_o \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad 5.7$$

$$x_o = xs\cos\theta - ys\sin\theta + d_x$$
$$y_o = xs\sin\theta + ys\cos\theta + d_y \qquad 5.8$$

$$\begin{bmatrix} x_o \\ y_o \\ 1 \end{bmatrix} = \begin{bmatrix} s\cos\theta & -s\sin\theta & d_x \\ s\sin\theta & s\cos\theta & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad 5.9$$

Figure 5-14 shows the results of applying translation, rotation and scaling on a rectangular shape.

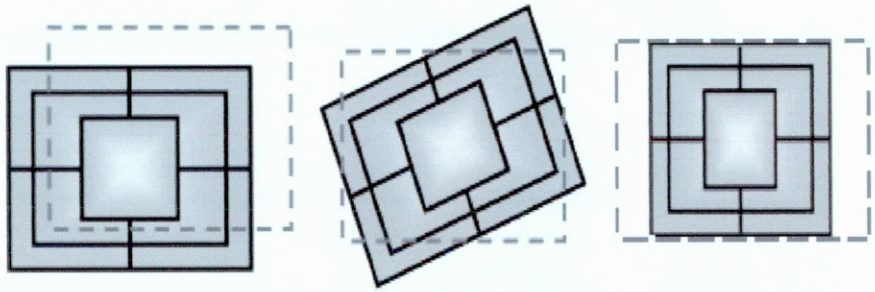


Figure 5-14: *Translation, rotation and scaling*

There are four parameters that determine the mapping between the images coordinates and the interactive panel coordinates namely; angle of rotation θ, scale factor S (s_x, s_y) and two displacements x_0, y_0. Rotation θ can be easily determined independently of the other parameters as follows:

$$\theta = \arctan((y_b - y_a)(x_b - x_a)) \qquad 5.10$$

Then two equations can be solved for s_X and s_Y:

$$s_x = \frac{w}{(x_b - x_a)}$$
$$s_y = \frac{l}{(y_b - y_a)} \qquad 5.11$$

where w and l are the image width and image height respectively.

Two further equations can then be used to solve for d_x, and d_y:

$$d_x = \frac{w}{2} - \frac{(x_b - x_a)}{2}$$
$$d_y = \frac{l}{2} - \frac{(y_b - y_a)}{2} \qquad 5.12$$

Figure 5-15 shows the application of image rectification using rotation, translation and scaling of an input image which contains an interactive panel marked with a black border on a bright background. In the rectified image in Figure 5-15(b), it is clear that the four corners of the interactive panel are located at the four corners of the image.

(a) The original image (b) The output image

Figure 5-15: *The results of image rectification using rotation, translation and scaling,*

Bilinear Warping for Four points

In the last section, it was assumed that the interactive panel will appear as a rectangle in the captured image frame. However, unless the interactive panel is precisely aligned parallel to the image frame, the resulting image will suffer from some perspective distortion. Image warping should therefore take in account correcting a quadrangle shape

rather than a rectangle. In this case, four control points will be needed to perform the warping. The four corners of the interactive panel can be mapped to the four corners of the image as shown in Figure 5-16.

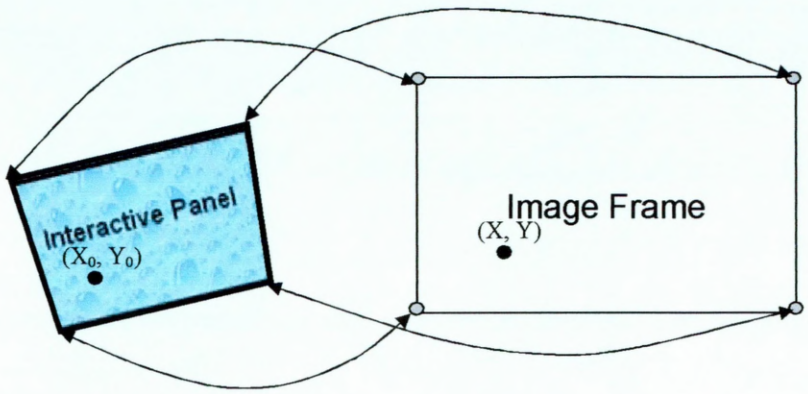


Figure 5-16: *Four points warping*

This type of mapping can be performed using the bilinear transformation formulated in the following equations:

$$x_o = c_1\, xy + c_2\, x + c_3\, y + c_4$$
$$y_o = c_1\, xy + c_2\, x + c_3\, y + c_4 \qquad 5.13$$

By assigning arbitrary coordinates to the corner pixels related to the interactive panel coordinates (e.g. (0,0), (0,1), (1,1), (1,0)), along with the image coordinates of the four corner points, it is possible to solve for the four parameters c_1, c_2, c_3, and c_4, since each point contains two independent values x and y.

Figure 5-17 shows the application of image rectification using the bilinear warping approach on an input image which contains an interactive panel marked with black border on a bright background. It is clear from the red rectangle (which has been added for verification) that interactive panel in the input image is not a rectangle.

It is also clear to see in the rectified image that the four corners of the interactive panel are located at the four corners of the image.

(a) *The original image* (b) *The resulting image*

Figure 5-17: *An example of image rectification using bilinear warping.*

5.3 Summary

This chapter has presented a panel normalisation technique which leads to a robust and stable environment even if the camera is tilted or the interactive panel slightly moved. It is argued that panel normalisation can work better when the user chooses an interactive panel that has a strong contrast levels with regard to its background to enable clear edge detection. An edge detection algorithm based on a sequence of steps was evolved to obtain the clear definition of the four edges. The resulting edges of the interactive panel then serve to define four corners, enabling bilinear warping based image rectification. Eventually, given the image coordinates of a point anywhere on the interactive plane (x, y), it is possible to invert the relationship and recover the coordinates of the point in relation to the interactive panel (x_o, y_o).

6 Change Detection for Toys Tracking and Turn Detection

6.1 Introduction

Chapter 5 has shown that the camera is maintained reasonably stationary where major changes in the frame images are caused by the toys and user's hand motions and that motion of the camera or the interactive panel can be corrected through the panel normalisation algorithm. These changes play a significant part in toy tracking and user action analysis.

This chapter will discuss how to detect changes in the frame image sequence according to the interaction modes, whether turn-based or real-time. Section 6.2 will present change detection, which will enable the segmenting of images into regions that represent toys. In addition, Section 6.2.5 will describe motion history as a reliable change detection technique for turn detection in a turn-based interaction mode. Then, in Section 6.3, these regions will be labelled (in the case of more than one toy having been moved in the same time interval) and their movements calculated in Section 6.4 to extract their statistical properties, such as dimensions, positions and orientations.

6.2 Change Detection

6.2.1 Overview

Tracking moving objects over time is a complex problem in computer vision and was an important research subject in the 1990s (Yang, 1992; Lowe, 1992; Coombs & Brown, 1992).

Furthermore, motion has been identified as a powerful feature for tracking moving objects, which can reveal the shape of moving object as well as other characteristics such as speed and function (Shapiro, 2001).

Shapiro and Stockman have identified four general cases of motion as follows:

- Still camera, single moving object and constant background

- Still camera, multiple moving objects and constant background
- Moving camera and relatively constant scene
- Moving camera and several moving objects

The simplest case is when a camera observes a relatively constant background. Moving objects across that background will result in changes to the image pixels associated with these objects. These pixels will form some regions (segmentations) with each region corresponding to one moving object. A common method for detecting changes in image sequences, where the camera is stationary, involves image subtraction to generate some regions of interest that represent the moving objects. This method is referred to as "Image differencing" (Rosin, 1997).

There were two approaches suggested in the literature to performing image differencing (Morries, 2004). These are:

Background Subtraction using the reference background image - In this method, the main issue is to obtain a good estimate of the background and then each frame image is compared (subtracted) with the background image. Various approaches can be used to select the background image; some of these approaches will be discussed in Section 6.2.2.1.

Frame Differencing between two frame images of the same scene at different times - This is performed by calculating the absolute values of the difference between the corresponding pixels in two subsequent frames in the image sequence. This has the advantage that a background image is not required as a reference image since only the last two frames are used for tracking. The disadvantage of frame differentiating is that some objects might become stationary for a period of time, which makes them disappear in the differenced image. Thus, only objects that have moved between the last two frames are tracked.

In this thesis, both background subtraction and frame differencing are used for change detection in the interaction modes as will be explained in the following sections.

6.2.2 Change Detection - Turn-Based Interaction Mode

In this thesis, a change detection approach was taken which combines "frame differencing" and "background subtraction" in order to keep track of all moving toys on the background, and between frames in the turn-based interaction mode.

One problem with image differencing is that there is a double region of each toy in the difference image, one where the toy was in the last image and the other region where it is in the current frame. One way to tackle this is to subtract the difference frame from a *reference image*, which represents the background (background subtraction).

Selecting the background image for background subtraction

There are various approaches used to obtain the background as a reference image. One method is simply to take a picture of the background without any moving objects. An alternative approach is to estimate the value of background pixels using a moving average. In this approach, the value of a particular background pixel is estimated as a weighted average of the previous values. Typically, pixels in the very distant past should be weighted at zero, and the weights increase smoothly (Forsyth & Ponce, 2003). The problem in image averaging is that objects that remain motionless for any period of time will be modelled as part of the background (Stauffer, 1998).

Stauffer& Grimson (Stauffer; 1998) and Huttenlocher (Huttenlocher; 1993) used probabilistic methods to select the background, where each pixel in the frame is modelled with a mixture of Gaussians. Incoming pixels are compared against the corresponding Gaussian mixture model and a match is sought. A match is defined as a pixel value within 2.5 Standard Deviations of a Gaussian. If a match is found, the parameters of the matching Gaussian are adjusted accordingly. However, if no match can be found, the least probable distribution is replaced with a distribution that models the incoming pixel. Gaussians that are more frequently matched are more likely to model background pixels and so incoming pixels are classified accordingly. This algorithm has since been enhanced to improve its learning rate and to account for object shadows (KaewTraKulPong & Bowden, 2001).

However, maintaining a Gaussians Mixture Model for every pixel is an enormous computational burden and results in low frames rates.

The approach taken in this thesis is straightforward and fast by simply setting up the camera, empty the scene of any moving objects and storing the background frame image as a reference.

The following algorithm presents the four main steps in achieving change detection in a turn-based mode while Figure 6-1 illustrates the inputs and the resulting images associated with each step. The algorithm starts by asking the user to capture the background image (B) at the beginning of the application. This can be done by removing

all the toys from the camera's view (the scene), so that the image from the camera represents the background condition.

Then, in the first step, the algorithm subtracts the two captured frames I_1 and I_2 to produce the "*silhouette image*" representing the difference in the image between the two frames. In step 2, this silhouette image is subject to a thresholding process to differentiate between noise (noise might be caused by illumination changes between the capture of the two frame images) and true differences. The threshold level (τ) is selected according to equation 6.1 as a linear combination of the mean (μ) and standard deviation (σ) of the silhouette image as follows:

$$\tau = k_1\mu + k_2\sigma \qquad 6.1$$

where K_1, K_2 are constants, A set of experiments with different values of the coefficients k_1 and k_2, has led to the adoption of the values $k_1 = 4$ and $k_2 = 2$. The resulting image I_{diff} contains two regions, where one region represents the moving toy and one is part of the background. Step 3 then performs the background subtraction on image I_{diff}. In step 4, all non-zero pixels are filled from the original frame image (I_2) to obtain the output image O and thus obtaining the region, which represent the moving toy.

Finally, the frame I_2 is stored as frame I_1. Then new captured frame from the camera will then be stored in I_2 and the process repeated.

Algorithm: Change detection for turn-based interaction mode

O	Output colour image
$k \in [1,3]$	RGB-colour channels
$i \in [0,W-1]$, $j \in [0,H-1]$	where W: image width, H: image height
τ	Threshold
B	Manual background capture
I_1	Automatic frame capture
I_2	Next frame capture

Repeat

Step 1 **Frame differencing**:

$$I_{diff1}(i,j,k) = |I_1(i,j,k) - I_2(i,j,k)|$$

Step 2 **Thresholding 1**:

$$I_{diff1}(i,j,k) = I_2(i,j,k) \quad if \quad I_{diff1}(i,j,k) > \tau$$
$$I_{diff1}(i,j,k) = 0 \quad Otherwise$$

Step 3 **Background Subtraction**:

$$I_{diff2}(i,j,k) = I_{diff1}(i,j,k) - B(i,j,k) \quad if \quad I_{diff1}(i,j,k) > 0$$

Step 4 **Output**:

$$O(i,j,k) = I_2(i,j,k) \quad if \quad I_{diff2}(i,j,k) \neq 0$$
$$O(i,j,k) = 0 \quad Otherwise$$

Copy $I_1 = I_2$

As already indicated, Figure 6-1 shows the input images and the resulting images associated with each step of the change detection algorithm in turn-based interaction mode. As can be seen from the figure, the moved toys are represented by regions. These regions will not then be changed until the user performs another task within the camera's view.

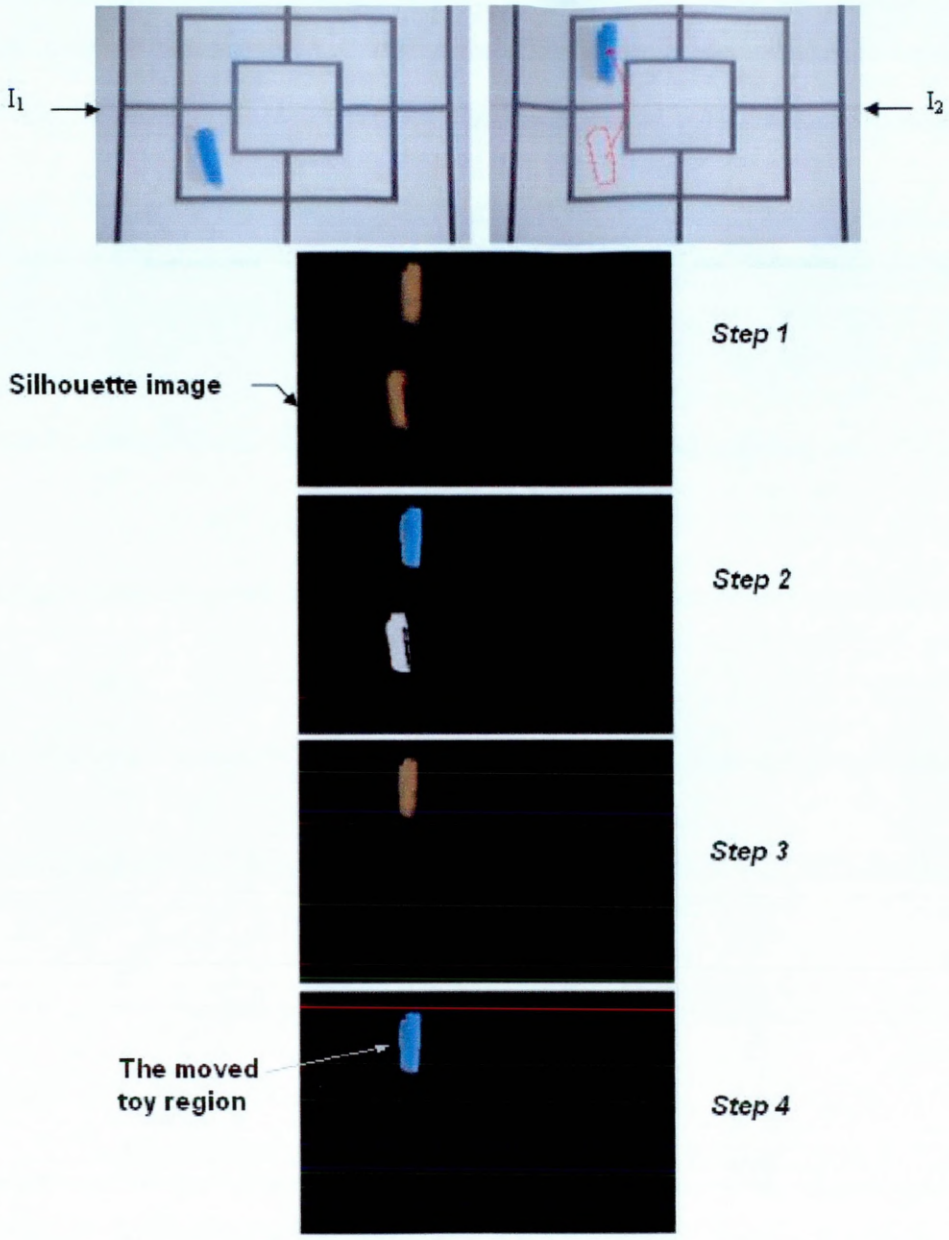


Figure 6-1: *The original images and the results images of the four steps of the change detection algorithm in the turn-based interaction mode*

If more than one toy has been moved at the same time, an image labelling process needs to be used in order to decide which region belongs to which toy. The image-labelling algorithm will be presented in Section 6-3.

6.2.3 Change Detection: Real-Time Background-Dependent Mode

In contrast to the turn-based interaction mode, the real-time-background-dependent interaction mode, users move toy or toys continuously in relation to the background. Therefore, background subtraction is adequate to track the moving toys. In addition, thresholding process is required to eliminate the noise caused by the illumination changes between the capture of the two frames (see Section 6.1.2 for details of the thresholding process). The following algorithm shows two steps of change detection for real-time-background-dependent mode. The algorithm obtains the background image B manually, and then subtracts it sequentially from each new frame *I*. Finally, the algorithm performs thresholding process for noise removing.

Algorithm: Change detection for real-time background-dependent mode

---

O The output RGB-colour image

τ Pre-defined threshold

B Manual background capture

$i \in [0,W-1]$, $j \in [0,H-1]$ where W: image width, H: image height

$k \in [1,3]$ RGB-colour channels

Repeat

New frame image I

Step1 **Background Subtraction**:

$$I_{diff}(i,j,k) = |I(i,j,k) - B(i,j,k)|$$

Step2 **Thresholding and the output**:

$$O(i,j,k) = I(i,j,k) \quad if \quad I_{diff}(i,j,k) > \tau$$
$$O(i,j,k) = 0 \quad Otherwise$$

Figure 6-2 shows an input image along with the resulting images following background subtraction and thresholding.

(a) Original image (b) Image after background subtraction (c) Image after thresholding

Figure 6-2: *An example of change detection for real-time interaction mode*

6.2.4 Change Detection: Real-Time Background-Independent Mode

In this mode, the changes between successive frames are tracked by using 'frames differencing' instead of 'background subtraction'. This means that the user does not need to store the background image as in the other two interaction modes.

The following algorithm shows the three steps making up the change detection algorithm for the real-time-background-independent mode. Each two frames I_1 and I_2 in sequence are differentiated and the thresholding process applied to obtain a segmented region that represents the moving toy (see Section 6.1.2 for thresholding process). Then the algorithm eliminates the small regions (false noise regions) caused by of the illumination changes between two captured frames. If the number of pixels C (of the segmented region) is less than a predefined value N, then the region is considered as a noise region and will be discarded.

N is identified by running the algorithm for five minutes on an empty background without moving toys and under different lighting conditions (e.g. bright and dark). The number of pixels C in each difference image is recorded and then N is calculated, as follows:

N = Average (C) 6.2

Algorithm: Change Detection real-time background-independent

O The output *RGB*-colour image

τ Pre-defined threshold

C The number of non-zero pixels in one silhouette image

N Noise Threshold

$i \in [0,W-1]$, $j \in [0,H-1]$ where W: image width, H: image height

$k \in [1,3]$ *RGB*-colour channels

Repeat

New frame image I_2.

***Step* 1 Frame Differencing**

$$I_{diff}(i,j,k) = |I_1(i,j,k) - I_2(i,j,k)|$$

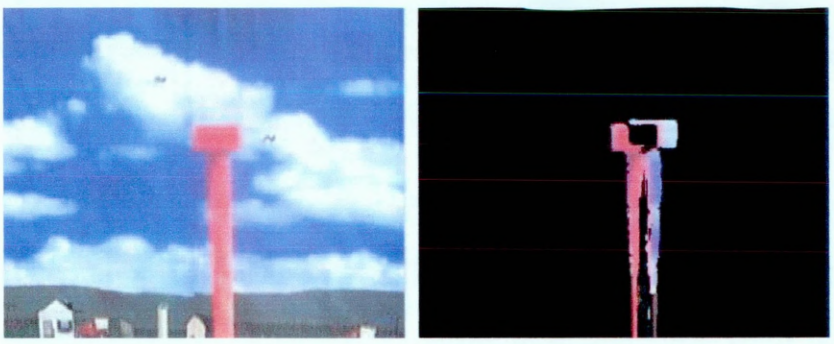
***Step* 2 Thresholding**

$$O(i,j,k) = I_2(i,j,k) \quad if \quad I_{diff}(i,j,k) > \tau$$
$$O(i,j,k) = 0 \quad Otherwise$$

***Step* 3 Elimination of pixels caused by jerky movements**

If $C < N$ the region will be discarded

Figure 6-3 shows the input and output images of the change detection process of the real-time background-independent mode where the number of pixels of the tracked region C is higher than the threshold N:



(a) Original image (b) Resulting image

Figure 6-3: *An example of change detection in real-time-background-independent interaction mode* C=65833 > N=200

6.2.5 Motion History Approach to Turn Detection

It was mentioned in Section 4.3.1 that a novel approach was developed in this thesis for turn detection in the turn-based interaction mode. This approach relies on the combination of hand and skin detection and motion detection techniques. This section will explain the operation of the motion history detection algorithm while section 7.5 will explain the hand and skin detection algorithm.

Users' interaction with the system causes major changes to a large number of pixel values in the frame sequence and usually takes at least 1 sec (i.e. δ motion duration) from start to end. These changes will be referred to as '*Major Motion*'. '*Minor Motion*' can then be described as slight changes in the pixels value over a short period of time. This can be caused by shadows or small objects movement.

Previous work has shown that motion over a frame sequence can be stored in a Motion History Image (*MHI*) (Davis & Bobick, 1997; Bradski, 2000). Every frame image, the *MHI* image is updated with the latest motions (changed pixels) for a period of time δ (typically one second). Pixels older than δ will then be discarded and new motion pixels added to the image. This method is referred to as Motion History (Bradski, 2000). In this dissertation, motion history has been utilized for detecting Major Motion, and thus for turn detection. An example of a *MHI* image is shown in Figure 6-4.

The following algorithm describes the four different stages involved in detecting Major Motion. The first step is to obtain the silhouette image (i.e. difference image) I_{Sil} as the result of frame differencing between a two-frame sequence I_1 and I_2. The silhouette image

I_{sil} is then used as a mask image enabling the assignment of the current timestamp value (t_s) (a floating point timestamp in the format: seconds.milliseconds) to those *MHI* pixels that have corresponding non-zero silhouette pixels. The *MHI* pixels older than ($t_s - \delta$) are also cleared if the corresponding silhouette values are 0. This procedure will create a layered representation of the motion in the *MHI*.

The most recent motion has the highest value and earlier motions decreasing values. Typically, this highest value is a floating point timestamp in milliseconds of the time elapsed since the application was launched. Finally, if the number of motion pixels is more than a constant threshold M, that means there is a Major Motion. Otherwise, there is a Minor Motion or no motion at all. M is defined by running the algorithm while small motions are caused by different types of things such as, fingers hair and shadow. Every frame the number of non-zero C pixels in the *MHI* image is recorded. Then M is calculated as follows:

$$M = \text{Average}(C) \qquad 6.3$$

Algorithm: Major Motion Detection

I_1 and I_2	Two frame images in sequence
I_{th}	Silhouette image that has non-zero pixels where the motion occurs.
MHI	Motion history image
δ	Maximal constant duration of motion track in milliseconds
N	Motion threshold
τ	Pre-defined threshold
t_s	Current time-stamp

t_s = Clock() / CLOCKS_PER_SEC;

where CLOCKS_PER_SEC=1000, clock () returns the number of clock ticks elapsed.

Step 1 **Silhouette Generation**

$$I_{diff}(x,y) = |I_1(x,y) - I_2(x,y)|$$

Step 2 **Thresholding**:

$$I_{th}(x,y) = 1 \text{ if } I_{diff}(x,y) > \tau$$
$$I_{th}(x,y) = 0 \quad \text{otherwise}$$

Step 3 **Update Motion History**

// update MHI and remove older motions after δ duration

$$MHI\ (x,y) = ts \quad if\ I_{th}(x,y) \neq 0$$
$$MHI\ (x,y) = 0 \quad if\ MHI = (ts - \delta)$$

Step 4 Check for a motion in a large number of pixels

Major Motion if C > M

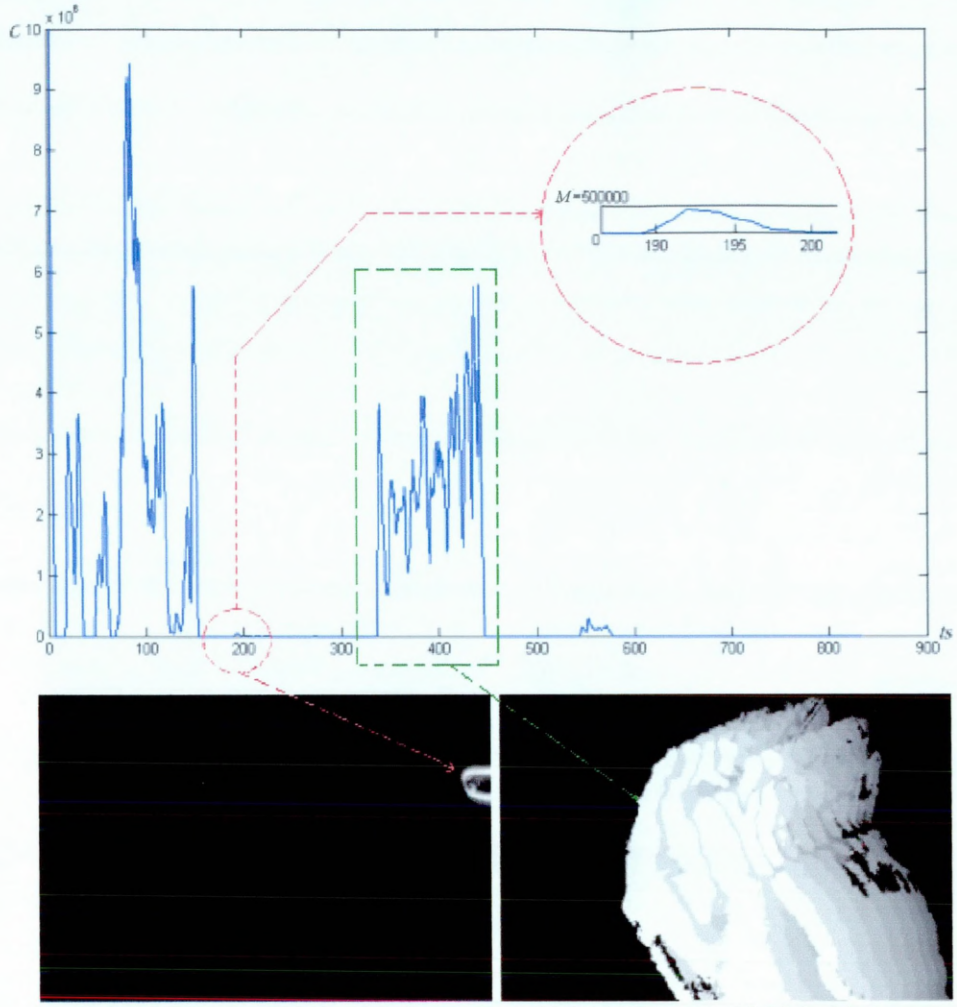
Minor Motion if C <= M

Figure 6-4 shows an input image with moving hand alongside an *MHI* image. In the *MHI* image, the current silhouette is in bright white, while past motions appear in dimmer shades of grey. This motion is considered as a Major Motion as the number of non-zero foreground pixels C is more than the threshold M. The red arrow in the *MHI* image shows the direction of the motion.

(a) Input image (b) Corresponding *MHI* image

Figure 6-4: *A hand moving a toy*

Figure 6-5 shows a histogram that illustrates a number of pixels C in motion in the Motion History Image *MHI* against the time-stamp *ts*. It is clear that when the user is performing a task, the number of pixels C in motion is either zero or less than the threshold M in the case of Minor Motion (see Figure 6-5(a)). Whereas C is rather high in the case of Major Motion (see Figure 6-5(b)).



(a) *An MHI image with* Minor Motion (b) An *MHI* image with Major Motion

Figure 6-5: *A histogram of an MHI image of turn-based interaction mode*

6.2.6 Labelling connected components

It has been found so far that change detection leads to regional segmentation. The number of segmented regions is the same as the number of toys moved in the time interval. The simplest case is when only one toy moves each time, which means that there will be only one segmented region per image.

If more than one toy is moved in the same time interval, then the corresponding regions have to be identified in order to become the inputs to higher-level procedures that

perform decision-making tasks such as toy positioning and identification. In the case of multiple toys, an important process that needs to be performed on images prior to region analysis is that referred to as "*Labelling Connected Components*". Labelling connected components assigns each region a unique label to separate it from all others. All the pixels within a labelled region are assigned the same label.

Different approaches were taken in the literature to performing fast and a simple connected-components labelling. The original labelling algorithm was developed by Rosnefeld and Pfaltz (Rosnefeld, 1966). It performs two passes through the image. In the first pass, the image is processed from left to right and top to bottom to generate labels for each pixel and all of its equivalent labels are stored in a pair of arrays, one containing all current labels and the other the minimal equivalent labels of those current labels. In the second pass, each label is replaced by the label assigned to its equivalent class.

Haralick (Haralick, 1981), and Lumia et al. (Lumia, 1983) have attempted to improving on the first method and removing the extra storage requirement for the pair of Arrays.
In addition, a "Recursive Labelling" algorithm was described by Shaprio and Stockman (Shapiro, 2001) assumes that the entire image can fit into memory, therefore it repeats calling a recursive function "search" to find 8-neighbours of each labelled pixel. This algorithm works well for small regions, while causing fatal memory errors when trying to label large regions.

In this dissertation, a fast technique has been developed for labelling connected components where the execution time is directly proportional to the size of the image. The following algorithm shows the developed labelling technique used to find the connected components of the 1-pixels (i.e. non-zero pixels) of a binary image I of width *w* and height *l* to produce a labelled output image LI.

The algorithm initializes the labelling image LI and negates the binary image I, so that all the 1-pixels become -1s (negative). This is essential to distinguish the unprocessed pixels (-1) from those of component with label 1. Then the process of finding the connected components becomes one of finding a pixel whose value is -1 in image I and 0 in image LI. When finding this pixel (-1 in I and 0 in LI), the algorithm assigns it a new label and jump to find its 8-neighbours, N_8(LB(i , j) that have value -1, (see Appendix B for 8-neighbour connected components).

If a neighbour is found, then assign it the same label and assign Check1 to 1. Repeat the process for these neighbours until all the connected components of this region have been

found and labeled. Then, if Check1 equal to 0 that means new connected components need to be found and labelled (label = label +1).

Algorithm: Label the connected components in a binary image

```
I        Original binary image
LI       The labelled connected component image.
w        Width of image I
l        Height of image I
label =1;
negate ( I );
initialize ( LI );
next_object:
  for i = 1 to w-1 {
     for j = 1 to l-1 {
     if ( I ( i , j ) == -1 and LI ( i , j ) == 0 )
     {
        LI ( i , j ) =label;
        goto labelling;
        }
     }
  }
Labelling:
Check1 =0;
 for i = 1 to w-1 {
     for j = 1 to l-1 {
     if (I ( i , j ) == 0 or LI ( i , j ) ≠ 0)
     continue;
 if N8 ( LI ( i , j ) ) == label;
 LI ( i , j ) = label;
 Check1 = 1;
     }
 }
 if Check1 == 1
 goto Labelling;
 else
 label = label + 1;
 goto next_object;
```

Figure 6-6 shows a binary image with three non-zero regions along with the result binary image resulting from the labelling of connected components. Three labels '1','2' and '3' are assigned to the pixels of the three regions respectively.

0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	1	0	1	1	1	0	0
0	0	0	0	0	0	0	0	1	1	1	1	0
0	0	0	0	0	0	0	0	1	1	1	1	0
0	0	0	0	0	0	0	0	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

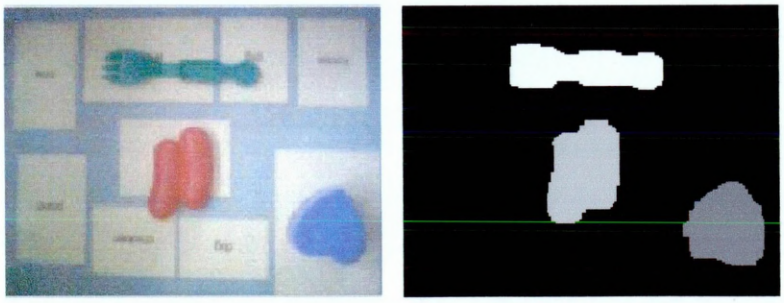
(a) Original binary image

0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	2	2	2	2	0	0	0	0	0	0
0	0	0	2	2	2	2	0	0	0	0	0	0
0	0	0	2	2	2	2	0	0	0	0	0	0
0	0	0	2	2	2	2	0	3	3	3	0	0
0	0	0	0	0	0	0	0	3	3	3	3	0
0	0	0	0	0	0	0	0	3	3	3	3	0
0	0	0	0	0	0	0	0	3	3	3	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

(b) Labelled binary image

Figure 6-6: *The application of labelling to connected components*

Figure 6-7 demonstrates the results of the labelling connected components algorithm on an RGB-input image.



(a) Original image (b) Resulting image

Figure 6-7: *An example of applying the labelling connected components algorithm to an image containing coloured objects*

6.3 Region Description using Moments

Image moments are generic and usually intuitive descriptors that provide a useful summary of region information and have been widely used to obtain region properties and to perform shape analysis, often for binary images (Hu, 1962; Dudani, 1977).
The calculation of the moments involves sums over all the pixels of an image or a defined region, and thus are robust against small pixel value changes. They can determine various attributes of segmented regions (e.g. region centroid and size) in a scene by inferring to the equivalent rectangle (or ellipse), which has the same moments as those measured in the object region (Horn, 1986).

In this thesis, image moments give efficient measures of a region's properties, namely:

1. The centroid (x_c, y_c) (i.e. centre of mass) which is the point used to specify a region's position
2. The lengths of the major axis w and the minor axis l, from the column axis of the region
3. The orientation θ,, which refers to how the region lies in the image plane

These properties can be calculated as follows:

The general two-dimensional $(p+q)^{th}$ order moments of a binary image I are defined as:

$$m_{pq} = \sum_{A} x^p y^q \tag{6.4}$$

where p & q = 0,1,2,.....

Let $I(x; y)$ be the image intensity at a position x, y, then the image moments, up to second order, are:

$$M_{00} = \sum_x \sum_y I(x,y) \qquad M_{11} = \sum_x \sum_y xyI(x,y)$$

$$M_{10} = \sum_x \sum_y xI(x,y) \qquad M_{01} = \sum_x \sum_y yI(x,y) \tag{6.5}$$

$$M_{20} = \sum_x \sum_y x^2 I(x,y) \qquad M_{02} = \sum_x \sum_y y^2 I(x,y)$$

Where M_{00} is the area of the region (i.e. the number of foreground pixels). The coordinates of the centroid c of a given region can be given as:

$$x_c = \frac{M_{10}}{M_{00}} \; ; \; y_c = \frac{M_{01}}{M_{00}} \tag{6.6}$$

The length l and the width w of the equivalent rectangle of the region are given by:

$$l = 4\sqrt{\frac{(a+c)+\sqrt{b^2+(a-c)^2}}{2}} \tag{6.7}$$

$$w = 4\sqrt{\frac{(a+c)-\sqrt{b^2+(a-c)^2}}{2}} \tag{6.8}$$

Where the intermediate variables a, b, and c are:

$$a = \frac{M_{20}}{M_{00}} - x_c^2 \; ; \; b = 2\left(\frac{M_{11}}{M_{00}} - x_c y_c\right) ; \; c = \frac{M_{02}}{M_{00}} - y_c^2 \tag{6.9}$$

Finally, the orientation θ of the equivalent rectangle is given by:

$$\theta = \frac{\arctan\left(\frac{2b}{a-c}\right)}{2} \tag{6.10}$$

The following three special cases need to be considered:

a) $b = 0$ and $a > c$

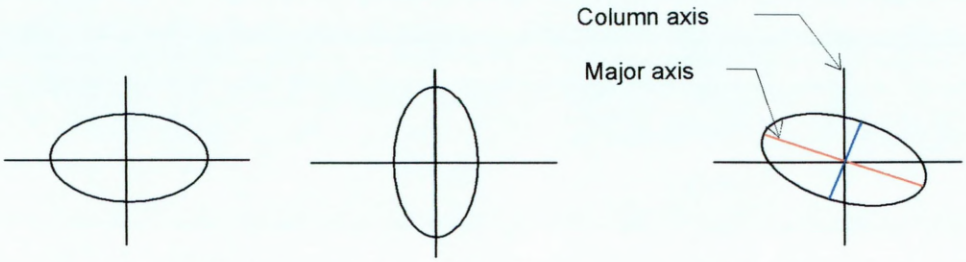
The major axis is oriented at an angle of -90° clockwise from the column axis and has a length of $4a^{1/2}$. The minor axis is oriented at an angle of 0° clockwise from the column axis and has a length of $4c^{1/2}$, (see Figure 6-8).

b) $b = 0$ and $a \leq c$

The major axis is oriented at an angle of 0° clockwise from the column axis and has a length of $4c^{1/2}$. The minor axis is oriented at an angle of -90° clockwise from the column axis and has a length of $4a^{1/2}$.

c) $b < 0$ or $a \leq c$

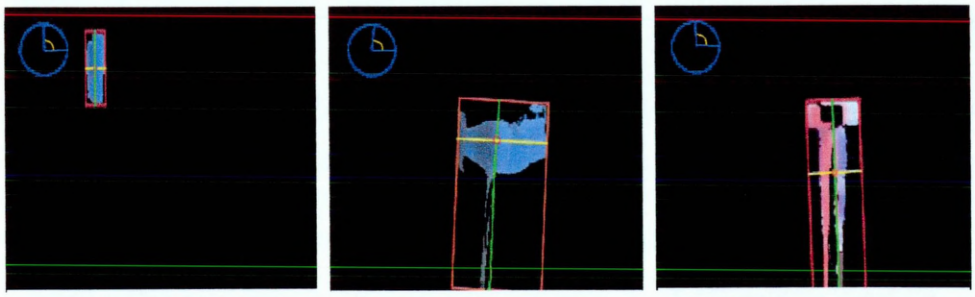
The major axis is oriented at an angle $-|\theta°|$ clockwise with respect to the column axis and has a length of w. The minor axis is oriented 90° clockwise from the major axis and has a length of l.



a) $b = 0$ and $a > c$ b) $b = 0$ and $a \leq c$ c) $b < 0$ or $a \leq c$

Figure 6-8: *Illustrations of three Special cases of the Moments intermediate variables.*

Figure 6-9 shows three example images of segmented regions along with the equivalent rectangles and the major and minor axes from the column axis of each region. The centroid of tracked toy is represented by a red circle, (a blue circle with a pointer shows the orientation).



(a) Turn-based mode (b) Real-time-background dependent mode (c)Real-time background independent mode

Figure 6-9: *Example images with the results of moment calculations in three interaction modes*

6.4 Summary

This chapter discussed and explained a number of algorithms, which are themselves the building blocks of the interaction mode algorithms presented in Chapter 4. In the course of this discussion, it was shown that image differencing was an essential approach for change detection in the three interaction modes and that:

- Background subtraction and frame differencing were combined to extract regions that represent moved toys in the turn-based interaction mode.
- Background subtraction was used to detect changes in the real-time background-dependent interaction mode.
- Frame differencing was used to detect changes in the real-time background-independent interaction mode.

In addition, image differencing was used to build a Motion History Image (*MHI*), which was then used in determining whether or not the user is performing a task within the camera's view as part of the turn detection process. In the case of multiple moved toys, the extracted regions were essential to be identified using 'labelling connected components' process.

Region descriptors such as dimensions, positions and orientations were obtained using the image moments method.

In the next chapter, algorithms related to colour recognition will be presented as a natural continuation of the work described in this chapter.

7 Colour Recognition

7.1 Introduction

In chapter 5, it was explained that object recognition is an essential technique for toy identification in the Interactive Toys Environment. This chapter extends the discussion from Chapter 6 to consider the use of colour within the object recognition structure and process.

Object recognition is an open and complex topic in computer vision and image processing. Colours, shapes and other object characteristics can be utilised to recognise an object. However, the object recognition problem was reduced in this thesis to that of colour recognition to identify toys of uniform colour. In addition, the concept of colour recognition was built upon the skin detection technique which is an essential requirement for turn detection.

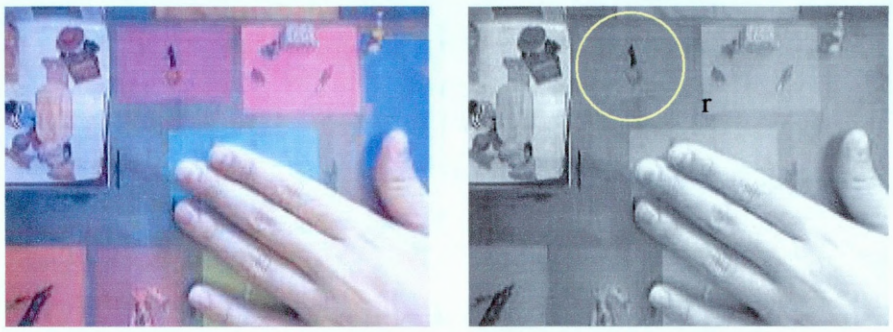
This chapter will give an overview of colour images, and how the use of colour spaces in particular, can be a powerful tool for colour classification (Section 7.3). The chapter will also consider the background to the image pre-processing used in this work to prepare the input colour images for colour recognition (Section 7.4) after which skin colour detection will be presented (Section 7.5). Finally, the chapter demonstrates the use of the colour recognition algorithm as a reliable and computationally inexpensive approach for toy identification.

7.2 Grey-Scale Images

A grey-scaled image is a digital image with a single intensity value per pixel, reducing the amount of data required by using one channel instead of three channels to represent an image. For example, the simplest conversion from colour-scaled image to grey-scaled image is formulated as the average pixel intensity of the three colour channels. This can be expressed as:

$$I(x,y) = \frac{r(x,y) + g(x,y) + b(x,y)}{3} \qquad 7.1$$

However, by converting colour images to a grey-scale image a considerable amount of information will be lost, in particular the edges information of objects in the image. Figure 7-1 shows a grey-scaled image from original colour-scaled image. Note that the boundaries between some of the regions are lost.



(a) *Colour image* (b) Resulting grey-scale image

Figure 7-1: *Conversion of colour-scaled imaged into grey-scaled image using averaging method*

7.3 Colour-Scale Images

7.3.1 Introduction

The previous section has shown that grey-scale images reduce the scene information content, which may be important for the application. The colour content of an image is therefore an important attribute for scene understanding and analysis. A colour image can be modelled as three image components (or channels) where each component corresponds to different colour. Numerically modelling colour in a three dimensional coordinates system is commonly referred to as "colour space". The notion of a perceptual colour space is to model the colour volume so as to better correspond to the way in which the human eye perceives colour and relative intensities. The colour spaces most often used in computer vision applications are RGB, Normalized RGB, HIS (HSV) and YCbCr spaces (Shapiro, 2001). These colour spaces will be explained as follows:

7.3.2 RGB colour space

The RGB colour space is the simplest and most common colour space and uses Red, Green, and Blue components. Using the 8-bit monochrome standard as a model, the corresponding colour image would have 24 bits (8-bits for each of the three colour channels).

RGB is mostly used for colour monitors and scanners. RGB makes three primaries (i.e. channels) for additive light, but not for subtractive. Additive means that colours are created by adding components to black (0, 0, 0). Subtractive means that light reflected off a surface (what the surface does not absorb). Therefore, for many applications, RGB colour information has to be transferred into a mathematical space that decouples the brightness information from the colour information.

Once this is done, the image information consists of a one-dimensional brightness (i.e. luminance) space and two-dimensional colour space, which does not contain any brightness information. However, it typically contains information regarding the relative amounts of the different colours. The RGB space and its corresponding colour cube can be seen in Figure 7-2 (RGB colour cube is 3-D illustration for RGB colour model). The origin represents black and the opposite vertex of the cube represents white. In the colour cube, red is (255, 0, 0), green is (0, 255, 0), and blue is (0, 0, 255).

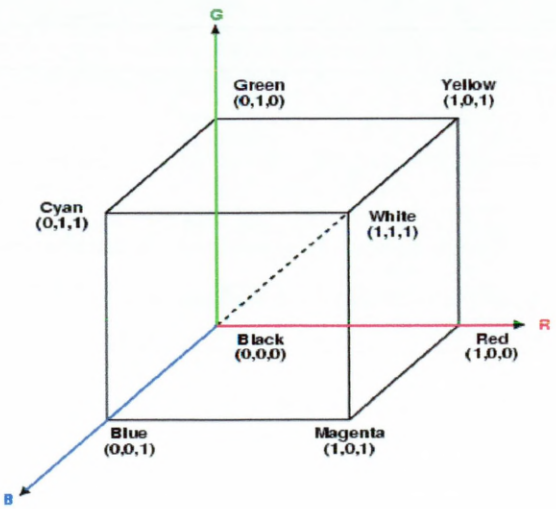


Figure 7-2: *RGB colour cube*

7.3.3 Normalised RGB colour space

The Red, Green, and Blue components of Normalised RGB (or Normalised-rg) space can be obtained from the three components of RGB space using the following formulations:

$$r = \frac{R}{(R+G+B)}$$
$$g = \frac{G}{(R+G+B)} \qquad 7.2$$
$$b = \frac{B}{(R+G+B)}$$

where r, g and b each lie in the range (0.0, 1.0).

These equations normalise the individual colour components to the sum of the Red, Green and Blue components. The three components of the Normalized RGB space are redundant because:

$$r + g + b = 1 \qquad 7.3$$

7.3.4 HSI colour space

The Hue-Saturation-Intensity (HSI) space encodes colour information by separating out an overall intensity value I from two values encoding chromaticity, namely Hue (H) and saturation (S). A related three dimensional representation is referred to as a 'Hexacone' and is shown in Figure 7-3. In the Hexacone, the angle around the hexagon is Hue, and distance from the centre axis is Saturation. Intensity is then plotted on the vertical axis. By increasing the intensity value, everything become less saturated and eventually becomes pure white. At zero intensity, everything fades to black.

The Hue-Saturation-Value (HSV) space is similar to the HSI space, using the term value instead of intensity. Determination of the HSI is computationally complex as is shown by Equations 7.3, where the Red, Green and Blue channels of RGB colour space are converted to Hue, Saturation and Intensity value:

$$I = \frac{1}{3(R+G+B)} \qquad 7.4$$

$$S = 1 - \frac{\min(R,G,B)}{I}$$

$$H = \cos^{-1}\left(\frac{(2R-G-B)}{2\sqrt{\left[(R-G)^2+(R-B)(G-B)\right]}}\right) if \quad B < G \qquad 7.5$$

$$H = 360 - \cos^{-1}\left(\frac{(2R-G-B)}{2\sqrt{\left[(R-G)^2+(R-B)(G-B)\right]}}\right) if \quad B > G$$

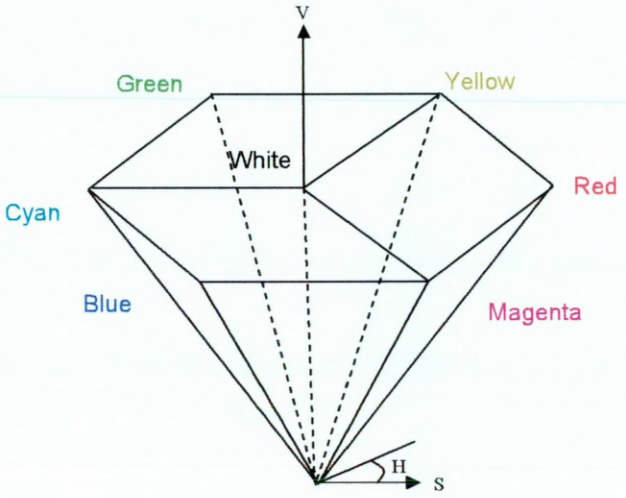


Figure 7-3: HSV *hexacone*

7.3.5 YCbCr colour space

The YCbCr colour space was established in 1982 by the International Telecommunications Union-Radio (ITU-R) as the basis for building digital video compression and processing algorithms such as JPEG and MPEG. In this colour space, luminance information is stored as a single component Y, and chrominance information is stored as two colour-difference components Cb and Cr. Thus, Cb represents the difference between the blue component and Y and Cr represents the difference between the red component and Y. The RGB to YCbCr colour space converter is designed to perform the following actions:

$$
\begin{aligned}
Y &= 0.299R + 0.587G + 0.114B \\
C_r &= R - Y = 0.701R - 0.587G - 0.114B \\
C_b &= B - Y = -0.299R - 0.587G + 0.886B
\end{aligned}
\qquad 7.6
$$

The rationale behind these coefficients can be found by considering that the luminance subtracted blue (B-Y) signal reaches its extreme values at blue (R=0, G=0, B=1, Y=0.114, B-Y=0.886) and yellow (R=1, G=1, B=0, Y=0.886, B-Y=-0.886). Similarly, extreme values of the luminance subtracted red (R-Y) signal reaches its extreme values at cyan (R=0, G=1, B=1, Y=0.114, R-Y=-0.886) and red (R=1, G=0, B=0, Y=0.299, R-Y=0.701). See Figure 7-4 for the YCbCr colour representation in relation to the RGB cube.

In practice, the Y value is encoded using more bits than the bits used for Cr and Cb because the human visual system is more sensitive to luminance than to the chromaticity values.

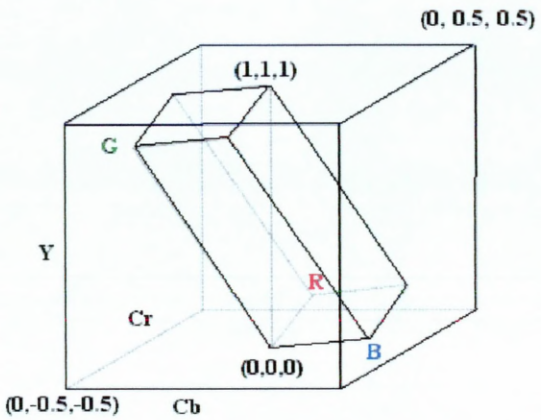


Figure 7-4: YCbCr *colour space cube in relation to the* RGB *colour space cube*

Table 7-1 shows the channel sequences and ranges of the four colour spaces RGB, HSI, Normalised-rg and YCrCb

Table 7-1: *Colour channel sequences and ranges*

Colour Space	Channel 1	Range	Channel 2	Range	Channel 3	Range
RGB	B	0 - 255	G	0 - 255	R	0 - 255
HSI	H	0 - 360	S	0 - 1	I	0 - 1
Normalized RGB	r	0 – 1	g	0 - 1	b	0 - 1
YCrCb	Y	0 - 255	Cr	0 - 255	Cb	0 - 255

7.4 Image pre-processing for colour recognition

Pre-processing techniques aim at enhancing specific image features or elimination of image information that is not required in order to prepare the image for further processing such as segmentation.

Image segmentation is the process of grouping the image into homogenous regions according to certain image characteristics such as colour. The essential difference between colour segmentation and colour recognition is that the former uses colour to separate objects without a priori knowledge about the image while the latter attempts to recognize colours of known colour characteristics. Although the two problems are the inverse of each other, results from segmentation can be useful in recognition (CVOnline, 2004).

There are different techniques for image pre-processing. The most common way is to smooth an image and convolve it with a spatial filter kernel that has the characteristics of the filter. Convolution (in an image-processing context) is a simple mathematical operation which provides a way of multiplying together two arrays, one of which is the original input image while the second is usually a smaller two dimensional array normally referred to as the 'Convolution kernel'. The convolution is performed by sliding the kernel over the image, generally starting at the top left corner, and then moving through all the positions where it fits entirely within the boundaries of the image. Let I be the original image and K the filter kernel. The convolution may be written as:

$$O(i,j) = \sum_{k=1}^{m}\sum_{l=1}^{n} I(i-k+1, j-l+1)K(k,l) \qquad 7.7$$

Where, i runs from 1 to M - m + 1 and j runs from 1 to N - n + 1.

Image Pyramids for image smoothing

The pyramid provides a hierarchical smoothing, segmentation, and hierarchical computing structure that supports fast analysis and search algorithms (OpenCV Manual, 2004).

Burt et al. (Burt, 1981) developed a new approach to enhance images with less processing time by using "*Image Pyramids*". An image pyramid usually defined as the representation of a digital image at different resolution levels (multiresolution). Each of these images, a smoother version of the original image at different scales and with fewer uniform regions. By choosing Gaussian kernel as a smoothing kernel for the image pyramid it is possible to get a set of low-resolution filtered images (the image Pyramid) which are referred to as a 'Gaussian pyramid'. Gaussian distribution has the form:

$$G(x) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad 7.8$$

Figure 7-5 represents a Gaussian distribution (Gaussian hump) with mean μ = (0, 0) and standard deviation σ=1

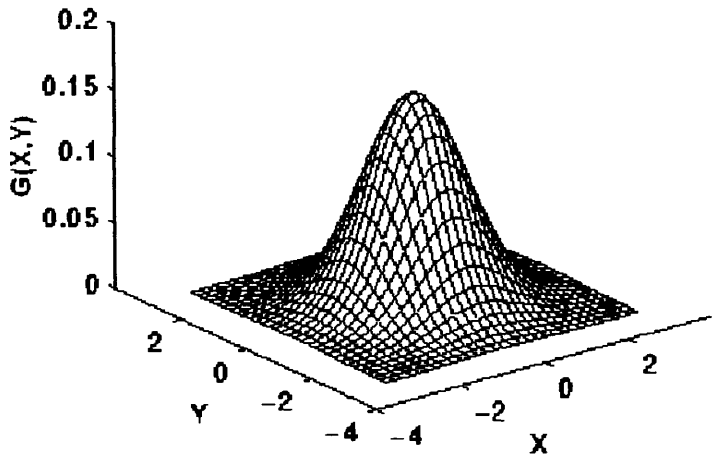


Figure 7-5: *Two-dimensional Gaussian distribution with mean (0, 0) and σ = 1*

The Gaussian pyramid has two important properties for image smoothing, these are:

1-'**Pyramid Down**' performs the down-sampling step of Gaussian pyramid decomposition. First, it convolves the source image with the Gaussian operator and then down-samples the image by rejecting even rows and columns. The convolved image is two times smaller than the source image. This process can be continued as far as desired or until the image size is one pixel. Figure 7-6 shows three steps of building a Gaussian Pyramid over a 240 x 180 image of a hand on a complex-coloured background.

Source image Level 1 Level 2 L3

Figure 7-6: *Results of applying three times 'Pyramid down' sampling method*

2-'***Pyramid Up***' performs the up-sampling step of the Gaussian Pyramid decomposition. First, it up-samples the source image by injecting even zero rows and columns and then convolves the resulting image with the Gaussian operator. Thus the destination image is two times larger than the source image. Figure 7-7, shows three steps of up sampling of a Gaussian Pyramid over the final output image shown in Figure 7-7.

Figure 7-7: *Results of applying three times 'Pyramid down' sampling method*

Two steps of pyramid up-sampling and down-sampling are utilised in this thesis to prepare the input frame images for skin detection and colour recognition as will be explained in the remainder of this chapter.

7.5 Detection of Skin Colour for Turn Detection

It was mentioned in Section 4.3.1 that a novel approach was developed in this thesis for turn detection in the turn-based interaction mode. This approach relies on the combination of skin detection and motion detection techniques. The following sections will present research on skin detection and show how skin pixels can be classified differently in a number of colour spaces. Then hand skin detection algorithm developed in this thesis will then be described in detail.

7.5.1 Skin detection

Skin detection has been gaining popularity and importance in the computer vision community, for example those by Zarit et al. (Zarit, 1999), Terrillon et al. (Terrillon 2000) and Brand & Mason (Brand, 2000). It can be defined as the process of selecting which pixels of a given image correspond to human skin involving a pre-process of colour space transformation and skin classification.

A simple method to build a skin classifier is to define explicitly the boundaries of skin cluster in some colour space, for example, Chai & Ngan (Chai, 1998), Fleck et al. (Fleck,

1996) and Jordao et al. (Jordao, 1999). This method needs to find both a good colour space and adequate decision rules to build a rapid skin classifier.

7.5.2 Selection of colour space for skin detection

Several colour spaces have been proposed in the literature for skin detection applications. Jones and Rehg (Jones, 1999) proposed skin / non-skin models based on distributions in RGB space for skin detection. A skin pixel classifier is devised through likelihood ratio approach, and a prior probability was determined based on threshold values. However, human skin colours differ in RGB space from person to person Yang and Ahuja (Yang, 1998). Moreover, high correlation between channels and luminance data make RGB a not very favourable choice for colour analysis and colour based recognition algorithms.

The HSV colour space has been proposed by Sobottka & Pitas and Saxe & Foulds (Sobottka, 1996; Saxe, 1996) because of the explicit discrimination between luminance and chrominance properties and also as it is more closely related to human colour perception. However, the shape of the human skin colour cluster in HSV space which is difficult to be represented mathematically.

In Yang and Ahuja (Yang, 1998) two components of the Normalized RGB colour space (r and g) have been proposed to minimize luminance dependence. However, skin colour covers almost all of the rg-colour space, which adds complexity to the skin detection problem.

Finally, YCbCr has been broadly used since the skin pixels form a compact cluster in the Cb-Cr plane. As YCbCr is also used in video coding and hence no transcoding is needed, this colour space has been used in skin detection applications where the video sequence is compressed (Albiol, 2000; Wang, 1997).

In order to decide which colour space achieves the best results for the Interactive Toys Environment, a training phase was undertaken to determine the nature of the hand pixels using three sample sets of hand images belonging to three ethnic groups; Caucasian, African and Asian. Hand skin images were collected from a top-down web camera under a variety of illumination conditions at different distances from the camera. Segmentation was applied to the samples and skin regions were selected manually and placed in one image "test-skin-image". Non-skin data such as rings, nail polish was manually removed. The "test-skin-image" then contained 3, 50, 000 skin pixels. The plot of the "test-skin-image" in R-G space is illustrated by the graph of Figure 7-8. This graph allows the visualization of the skin colour along the R and G axes. It is clear that a single cluster is

generated by the samples. However, these samples occupy a relatively large range of the total colour space.

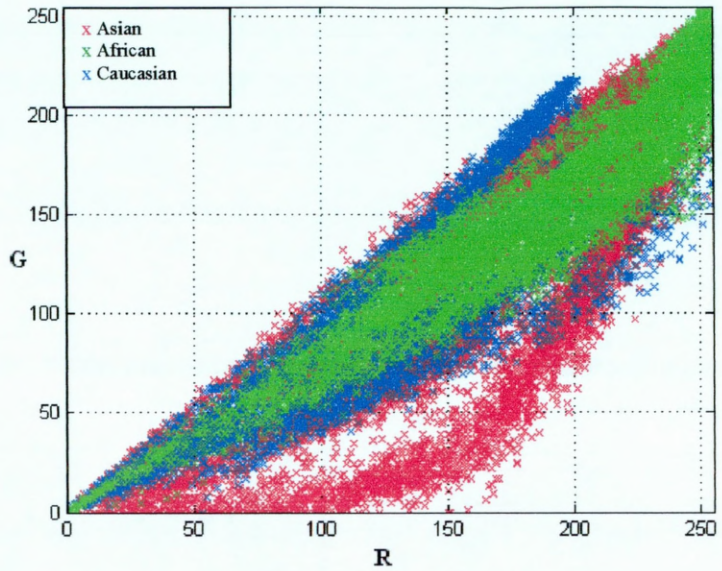


Figure 7-8: *Human hand skin samples plotted in* RG *space*

Each sample pixels in the "test-skin-image" were then mapped into Normalized rg-colour space. The plot of the "test-skin-image" in rg-colour space is illustrated by the graph of Figure 7-9. It is possible to see from the graph that the skin colour tends to cluster on three areas of the rg-colour space; one cluster for Caucasian, one for African and another for Asian. This causes a searching complex for the skin colour which covers almost all of the rg-colour space.

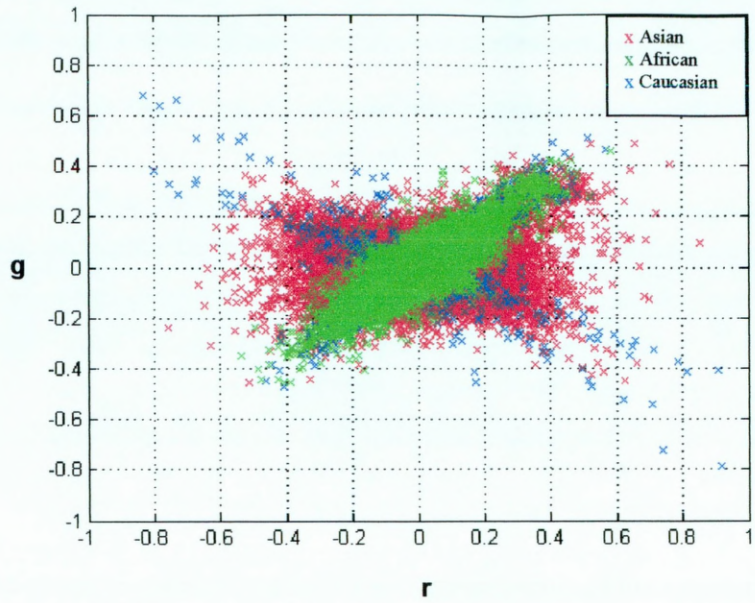


Figure 7-9: *Human hand skin samples plotted in* Normalised-rg *space*

Each sample pixels in the "test-skin-image" were then converted into HSV space, the luminosity was discarded and the results focused on the Hue-Saturation-space (HS space). The HS components are plotted in polar coordinates in Figure 7-10 where the HS space shows a tighter cluster. Hue being the angle θ ranging from 0 to 360 with red at θ = 0 and Saturation ranges from 0 to 1. The human skin colours cover almost all the S values for a limited H range. This leaves the S range ineffective in detecting the human skin, and makes it difficult to detect the skin colour based on the H by itself.

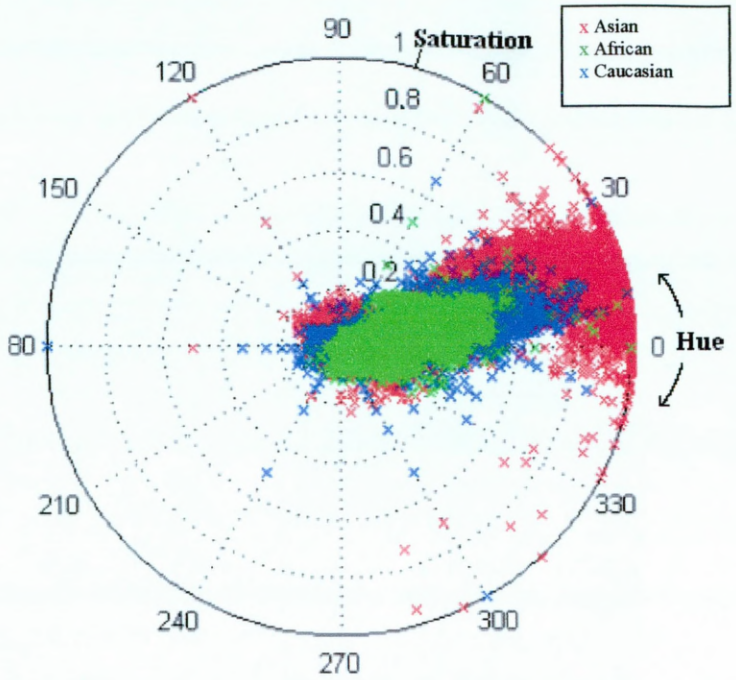


Figure 7-10: *Human hand skin samples plotted in* HS *space*

Finally, the sample pixels in the "test-skin-image" were converted into YCbCr form and the luminosity Y discarded in order to focus on the two-dimensional space Cb-Cr. The results can be seen in graph of Figure 7-11.

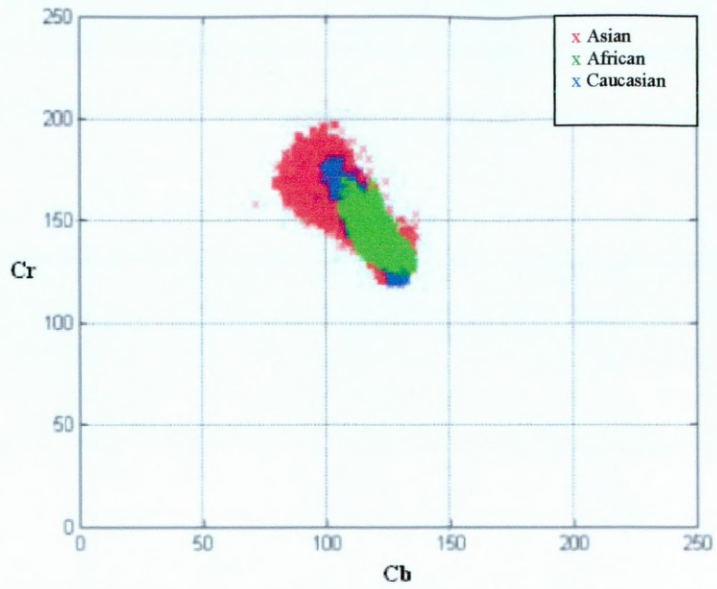


Figure 7-11: *Human hand skin samples plotted in* Cb-Cr *space*

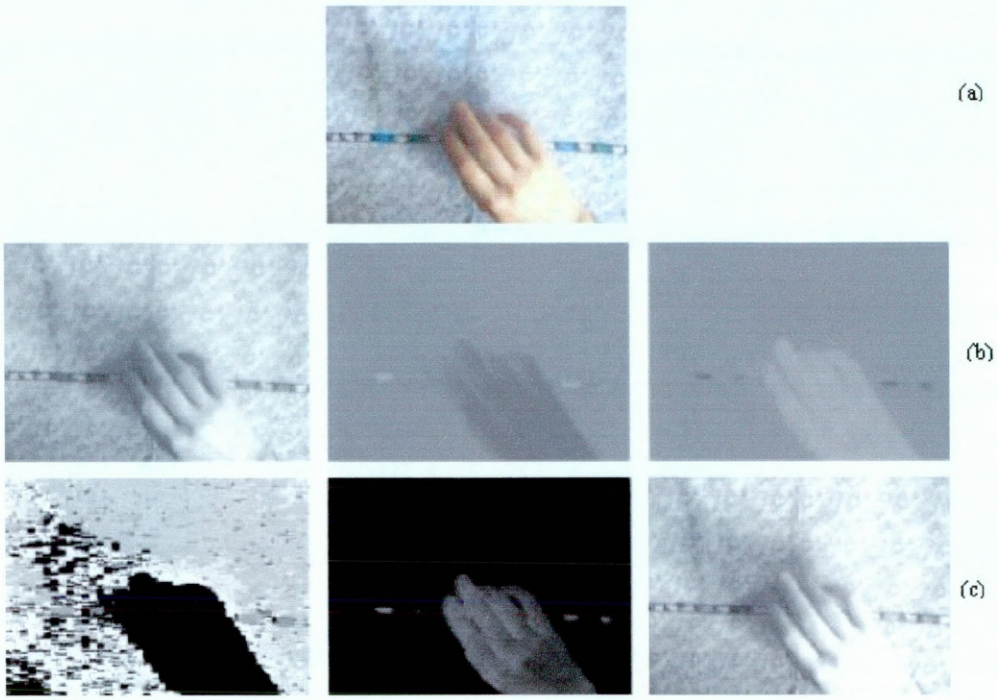
The skin colour pixels for Caucasian, African and Asian subjects were narrowly distributed and formed a compact cluster in the Cb-Cr plane. This cluster has minimum and maximum values over the Cr and Cb axes which will be referred to as the 'R_{cr} and R_{cb} skin ranges'. The respective ranges R_{cr} and R_{cb} can be obtained from the Cb-Cr plot shown in Figure 7-11 as follows:

$$R_{cr} = [138,178]$$
$$R_{cb} = [77,137] \qquad 7.9$$

Figure 7-12(a) shows a colour image with a user's hand moving an object on a bright background. In Figure 7-12(b) this image is converted into YCbCr with each channel (Y, Cr and Cb) shown as a grey-scale intensity image. In Figure 7-12(c) the same input image is converted into HSV colour space and each channel (Hue, Saturation and Intensity) is again shown as a grey-scale intensity image.

Although the saturation image provides a clear segmentation between the skin and other areas of the image, it is difficult to segment the skin in the Hue channel. It can be seen that combining both Cr and Cb images provides a clearer segmentation between the

hand's skin and the background. Also, Cr and Cb do not carry brightness information which means that this method will work in varying lighting conditions.



(a) An RGB image with a user's hand moving a toy

(b) The separate channels of the corresponding YCbCr image

(c) The separate channels of the corresponding HSV image

Figure 7-12: *Hand detection*

7.5.3 Hand skin detection Algorithm

The aim of skin detection as considered in this dissertation is to classify the pixels of the input image into skin-colour and non-skin-colour pixels and then to decide whether the image contains a hand (and on some occasions a foot) or not. Thus, segmenting and recognising the hand as an image feature are not required.

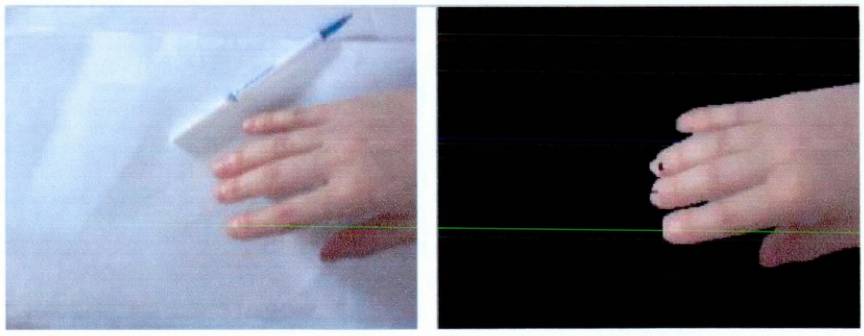
Every new frame image (I) is translated from RGB into the YCbCr colour space. Pixel $I(x, y)$ is classified as skin colour if both its Cr and Cb values fall inside the respective

skin ranges R_{cr} and R_{cb}; otherwise, the pixel is classified as non-skin colour and will be set to zero. The output of the colour segmentation is then stored in a mask image represented by:

$$Mask_{skin}(x,y) = \begin{cases} 1 \, if [Cr(x,y) \in R_{Cr}] \cap [Cb(x,y) \in R_{cb}] \\ 0 \qquad\qquad otherwise \end{cases} \qquad 7.10$$

The number of skin pixels are then counted. If the result image $Mask_{skin}$ has a number of skin pixels greater than a threshold value N, then the image will be considered as a skin image. After a number of tests with different users interacting with the Interactive Toys Environment, it was found that the threshold level for giving the best results was with N having the value 100,000. However, N can be set by the user according to the final application. If the user is intending to use bigger objects (bigger than the size of the hand) with a colour similar to that of skin, then N needs to be set to a higher value.

Figure 7-13 shows an input image with a hand and bright background along with the resulting image for skin detection using YCbCr colour space. In addition, some morphological operations such as erosion and dilation are used in this example image to fill some of the holes and to remove noise to improve the figure.



(a) Original image with a hand on a bright background

(b) Resulting image

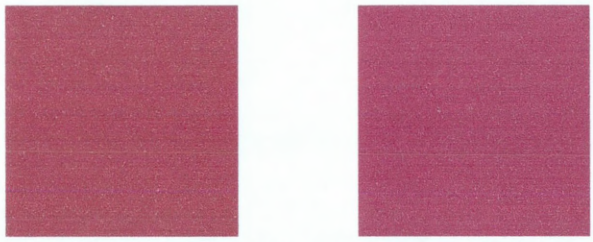
Figure 7-13: *An example of the application of hand skin detection*

7.6 Toy Colour Recognition

For simplicity and time efficiency, an algorithm for recognizing uniform-colour toys was developed in this thesis with similar approach to that for skin detection using YCbCr. The YCbCr colour space is chosen over RGB, Normalised-rg and HSV for the following reasons:

1- RGB and Normalised-rg are not selected for the same reasons as set out in Section 7.5.3.

2- HSV is not selected because it relies on the Hue component (H) to perform colour recognition. Figure 7-14 shows two colours which look very similar with the same saturation component S = 0.5245. However, the absolute difference of their Hue component is very big ($\triangle H = 0.9765$).

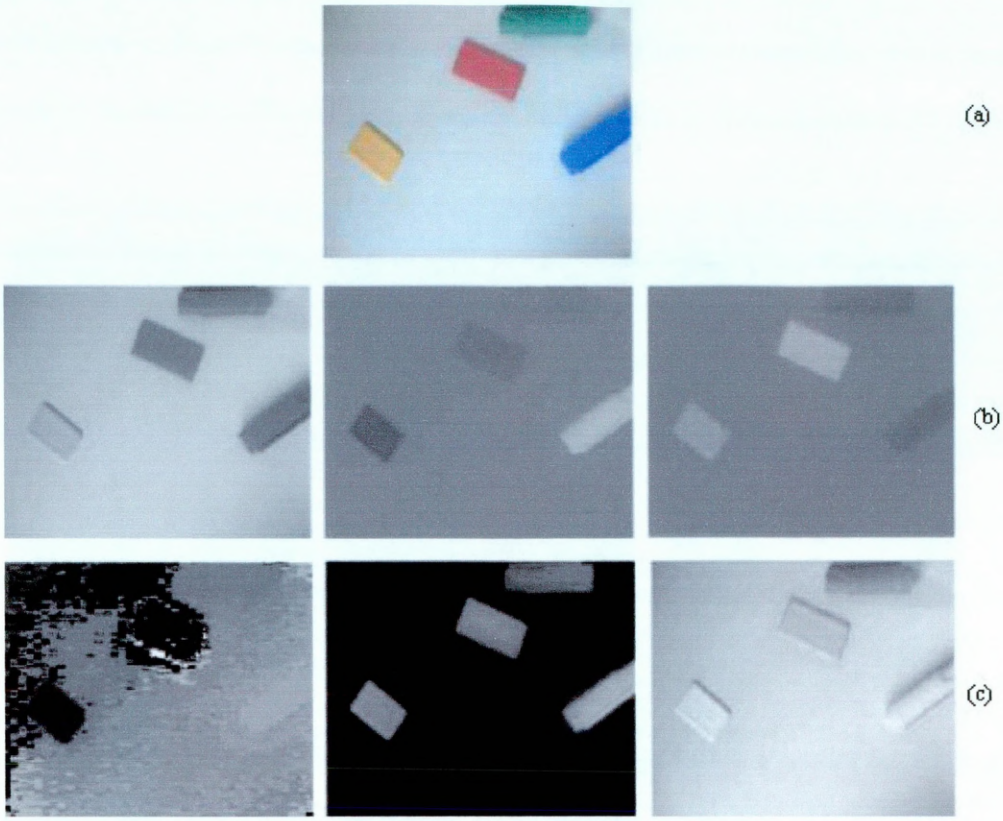


(a) H=0.0069 S=0.5245, V=0.5608 (b) H=0.9844 S=0.5245 V=0.5608

Figure 7-14: *An example of how similar colours have very different* H

Figure 7-15(a) shows a colour image with four colour objects red, green, blue and yellow on a bright background. This image is converted into YCbCr form and each channel (Y, Cr and Cb) shown as greyscale intensity image in Figure 7-15(b). In addition, the same input image is converted into HSV colour space and each channel (Hue, Saturation and Intensity) shown as a grey-scale intensity image in Figure 7-15(c).

In Figure 7-15, it is difficult to distinguish any colours in the Hue channel, even the boundaries between the objects and the background are not clear. In the Saturation channel, it is possible to segment the image, however, it is difficult to distinguish between the colours. It is also clear to see that distinction between colours is higher when combining both Cr and Cb channels. For example, green and blue objects look similar in the Cb channel and different in the Cr channel.



(a) An input colour image

(b) The separate channels of the same image in YCbCr format

(c) The separate channels of the same image in HSV format

Figure 7-15: *Colour detection*

Similar to the skin detection, a training procedure was performed in order to verify that YCbCr colour space is suitable for toy colour recognition. Therefore, 288 sample images of blue, red, green and yellow colours (72 sample images of each colour) were collected from a standard web-camera under varying and reasonable lighting conditions. Segmentation was applied to them and colour regions manually selected. Then the RGB values from these regions were gathered.

Let I_r, I_g, I_b, I_y be four images, each image containing 72 samples of selected colours red, green, blue and yellow (other colours can also be chosen). These images were then translated into YCbCr colour space when the colour pixels form a compact cluster in the Cb-Cr plane. By using a similar approach to that for skin detection, it was then possible to

obtain the respective range for each colour. Colour pixels were identified by the presence of a certain set of Cb and Cr values narrowly distributed in the YCbCr colour space.

Figure 7-16 shows a two-dimensional plot of Cr-Cb taking in account that Y describes the luminance values. It is clear from the plot that the red, green, blue and yellow colours were narrowly distributed in the plot over four regions.

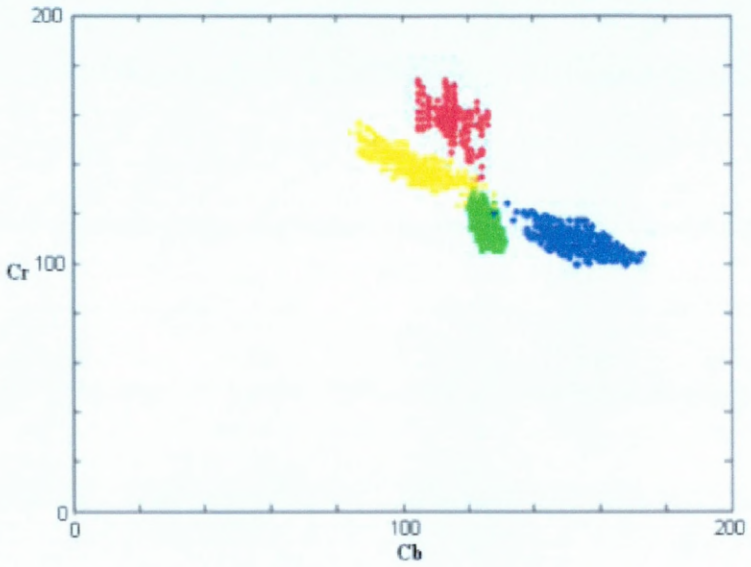


Figure 7-16: *Colour distributions of* Cr *and* Cb

The R_{cr} and R_{Cb} colour ranges of each colour can be obtained by finding the lowest and highest values of the Cr axis and similarly the lowest and highest values of the Cb axis. Table 7-2 shows the respective R_{cr} and R_{Cb} ranges of the Cr and Cb values that correspond to the red, green, blue and yellow colours respectively.

Table 7-2: *The* R_{cr} *and* R_{cb} *values as the respective ranges of red, green, blue and yellow colours*

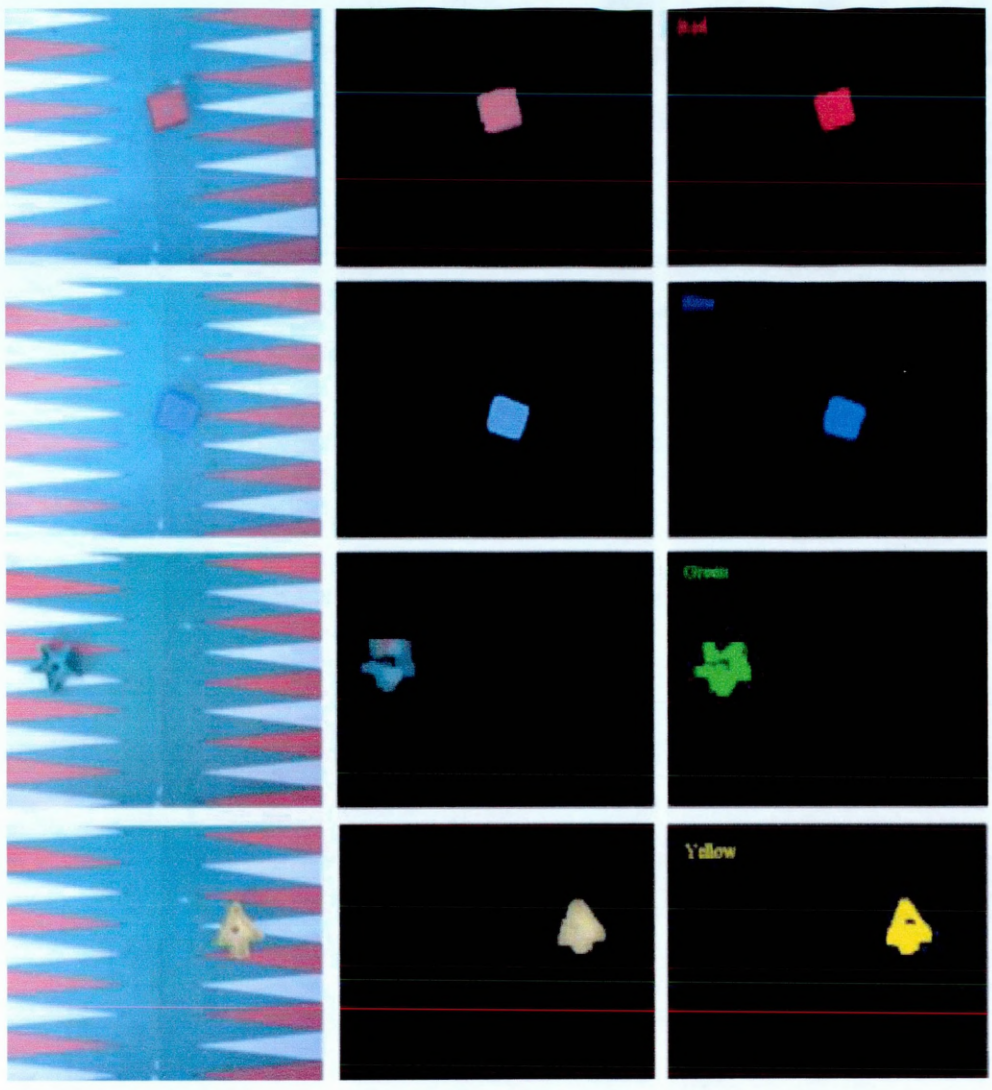
	R_{cr} Low	R_{cr} High	R_{cb} Low	R_{cb} High
Red	136	176	104	128
Green	104	130	120	130
Blue	100	124	134	174
Yellow	130	162	84	128

With these ranges, colour recognition algorithm in this thesis can recognise uniform-coloured toys as follows. Consider an input image (I) in YCbCr format after converting it from RGB. Then, the pixel I(x, y) is classified as red colour pixel if both its Cr and Cb values fall inside their respective red ranges, R_{cr} and R_{cb}, otherwise the pixel classified as non-red pixel. The output image O can be found as follows:

$$O(x,y,k)=\begin{cases}(255,0,0) & if \quad [136\le Cr(x,y)\le 176]\cap[104\le Cb(x,y)\le 128] \\ 0 & otherwise\end{cases} \qquad 7.11$$

where k∈1-3

The same process is followed for the other colours. The result of the colour recognition algorithm is four mask images, one for each colour. This algorithm can be easily extended to any other colours. Figure 7-17(a) shows original four images with red, blue, green, yellow toys on a board game. Figure 7-17(b) shows the result segmented regions of these toys after change detection and labelling respectively while Figure 7-17(c) shows the resulting colour mask images of the original images.



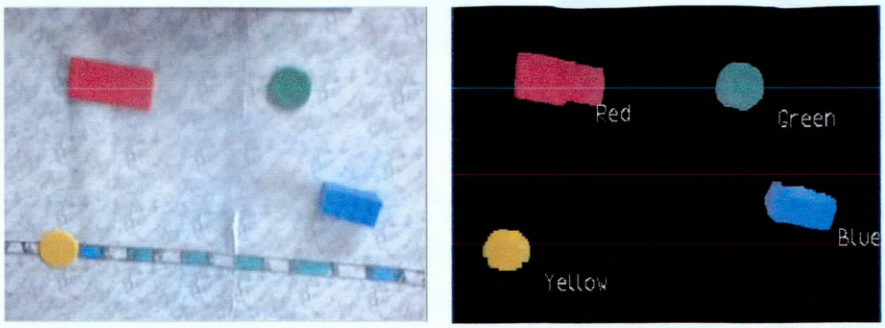
(a) Example images of uniformly coloured red, blue, green and yellow toys

(b) Resulting images following change detection and region segmentation

(c) Resulting (mask) images following colour recognition

Figure 7-17: *Four examples of colour recognition test results*

Figure 7-18(a) shows multiple coloured toys on a bright background while Figure 7-18(b) shows the output following colour recognition with four segmented regions.



(a) Original colour image (b) Resulting image

Figure 7-18: *An example of colour recognition results for multiple colour objects*

7.7 Summary

This chapter has presented a robust and fast method for recognising toys of four colours (red, green, blue and yellow) and detecting hand skin colour under vary lighting condition. Several colour spaces were discussed and the benefits of colour spaces demonstrated. The YCbCr colour space was chosen over several alternative colour spaces due firstly to its capability to decouple the brightness information from the colour information, secondly to its simplicity as it requires three linear equations to transform from RGB and thirdly most of the tested colours formed compact clusters in the Cb-Cr plane.

It has been also demonstrated that colour is a powerful feature capable of facilitating the robust tracking of uniform colour toys and skin in its own right.

8 Discussion and Evaluation

Introduction

Performance evaluation of computer vision systems is an essential task, and one which cannot be separated from the design process. However, it is important that these systems are designed for testability by adopting a methodology within which performance criteria can adequately be defined (Courtney and Thacker, 2001).

This chapter discusses and evaluates computer vision algorithms presented in Chapters 5, 6, and 7 in terms of accuracy and computational time. In addition, this chapter will identify the limitations of such algorithms and give some technical suggestions for future enhancement.

Chapter 9 will then present several applications which were developed in the course of this work. These applications will support the evaluation process by demonstrating the robustness and usability of the entire Interactive Toys Environment.

The rest of this chapter is organised as follows. Section 8.2 describes the framework used in the thesis to evaluate the performance and behaviour of computer vision algorithms using both manual ground truth and empirical testing. Section 8.2 also presents the results of these evaluation tests supported by examples.

Section 8.3 presents the computational efficiency of both computer vision algorithms and interaction modes. Finally, Section 8.4 presents some concluding remarks for the chapter.

Experimental Results for Error Evaluation

In this work, the experimental setup consists of a Logitech Pro 4000 web camera of 320x240 resolution providing 40° field of view and 15/30 frames per second (See appendix C for tests on web cameras). This camera is connected to a USB 2.0 port on a Pentium 4, 1.4 MHz PC.

All the test images are recorded during real-time interaction with the Interactive Toys Environment. There was not, to our knowledge, a standard image dataset (i.e. benchmark dataset) to evaluate this work (e.g. dataset of hands on a flat surface and under different illumination conditions).

8.2.1 Evaluation of Panel Normalisation

Evaluating panel normalisation is a challenging task due to its use of a large number of image processing techniques such as edge and corner detection. Therefore, a subjective evaluation of the panel normalisation algorithm will be presented in this section, acknowledging a number of limitations.

A number of trials were carried out with the panel normalisation algorithm under a various illumination conditions and using 15 different interactive panels. The algorithm performed very well when the interactive panel was located completely within the camera's view with clear contrast to its background and in ideal case of day light. Figure 8-1 shows three pair of images taking from the test frame sequence using arbitrary rectangle-shaped panels, which have clear contrast to their backgrounds in ideal day light. It possible to see from these images that the mapping between the interactive panel to the image pixels was successfully achieved.

(a) Original images (b) Normalised images

Figure 8-1: *Panel normalisation tests on three different types of interactive panel*

However, there are some limitations on the panel normalisation algorithm as follows:

1) **False normalisations**:

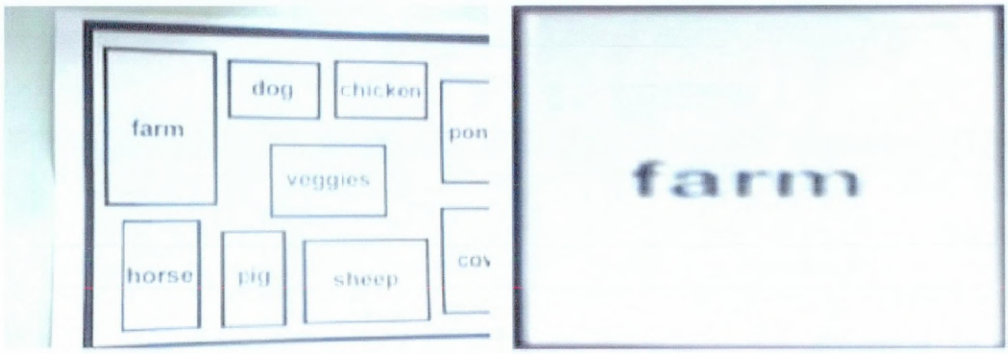
- Confusion might happen when the user rotates the main axes of the interactive panel through more than a 45° angle. This might cause a spurious mapping between the panel's points and the image's pixels as shown in Figure 8-2.
- Confusion might also occur when panel normalisation rectifies an internal rectangle instead of the panel's external boundary as shown in Figure 8-3.

(a) (b)

(a) Input image containing an interactive panel in which the main (horizontal) axis has been rotated through a 55° angle from the main axis of the image

(b) Resulting image showing a 90° rotation from the desired position

Figure 8-2: *Effect of excess rotation on panel normalisation*



(a) (b)

(a) Input image containing an interactive panel partially out of the camera's view

(b) Resulting image showing the spurious normalisation of an internal rectangle

Figure 8-3: *Panel normalisation on an internal rectangle*

2) **Errors in locating one or more of the interactive panel's corners**:

- The interactive panel might have a reflective surface which does not allow its boundary to be clearly defined as shown in Figure 8-4.

- The interactive panel might be partially out of the camera's view as shown in Figure 8-5.

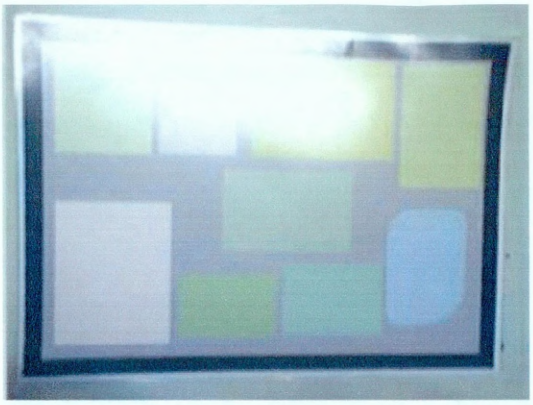


Figure 8-4: *Interactive panel with bright areas in the top-left- corner*

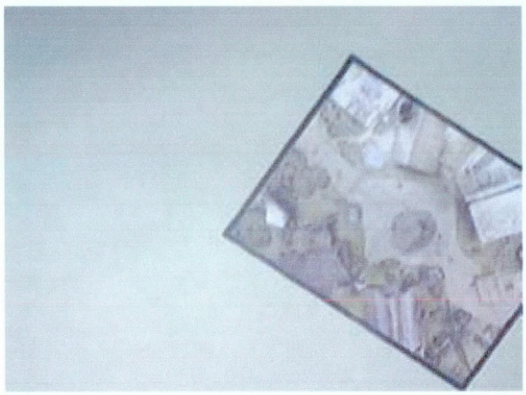


Figure 8-5: *An interactive panel that is partially out of the camera's view*

8.2.2 Evaluation of Toy Tracking

The approach to evaluating the performance of the detection and tracking system typically uses a "ground truth"[11] to provide independent and objective data (e.g. classification, location and size) that can be related to the observations extracted from the

[11] Ground truth is an estimate of what is thought to be in the test data. It may be determined by an independent method, for example using standard image test set, or it may be manually defined (Courtney, 2001).

video sequence (Black, 2003). The aim of this section is to present the approach taken in this thesis in evaluating the performance of the toy tracking algorithm and also to illustrate the associated results. The evaluation was performed by quantitatively comparing the tracked positions of the toys resulting from the change detection algorithm (which was presented in Chapter 6) against the ground truth positions.

Establishing the testing data and ground truth

Tracking data, which are a combination of ground truth positions and test tracked position, were generated during a turn-based interaction process according to the following four tracking scenarios:

Illumination changes - Five datasets were generated under different conditions of illumination as follows:

- Presence of shadow caused by objects out side the scene
- Neon light
- Ideal day light [12]
- Desktop lamp light
- Dark (for example cloudy day)

Toy sizes - One dataset was generated after testing with large-scale toys of four colours (the toy is considered as of large-scale if it occupies approximately $1/20^{th}$ of the area of the image).

Differing background complexities - Three datasets were generated using thirty images recorded while interacting with simple blue toy on a range of different backgrounds. These were:

- Light grey background
- Blue background
- A complex background made up of a variety of different colours and textures

Translation - One dataset was generated from thirty images recorded while moving a single red toy into different positions within the camera's field of view in order to check lens distortion (some camera lenses can introduce distortions such as a curvature in the

[12] Ideal day light is obtained in a room illuminated with normal day light.

straight lines around the perimeter of the image. Figure 8-6 shows two types of distortion that might be encountered).

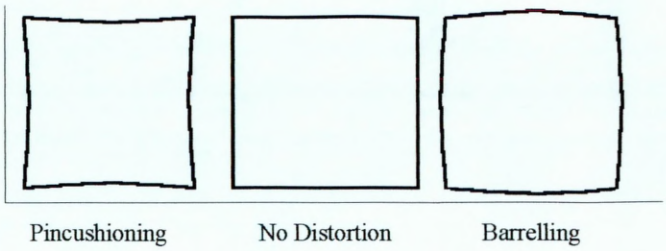
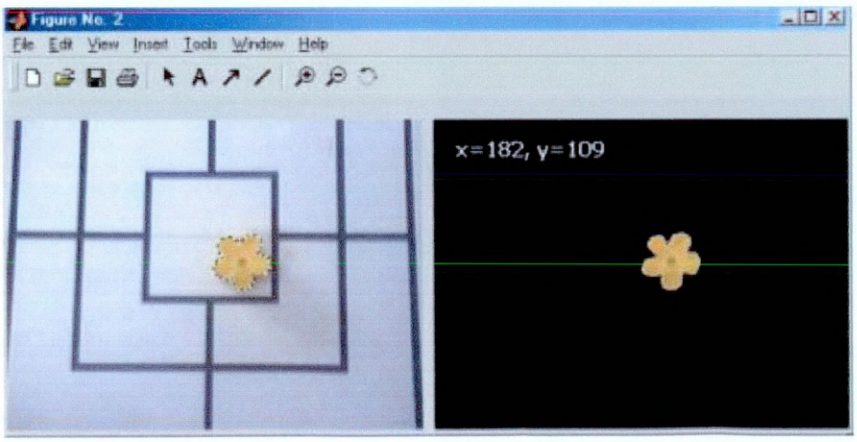


Figure 8-6: *Examples of lens distortions*

Generating the Ground Truth Data

In this thesis, MatLab software is used to produce the ground truth data. Every frame image is analysed and processed by manually segmenting the foreground regions (the toys regions) and then calculating the positions of the centroids of these regions.

Figure 8-7 shows a MatLab figure consisting of an input image and the resulting ground truth image containing the centroid data after manually selecting the toy region using MatLab selection tool.



(a) Input image (b) Ground truth image

Figure 8-7: *MatLab figure used to generate ground truth data*

Defining the error metric

Evaluation of toy tracking performance was carried out by defining a coherence error metric that indicates the level of agreement between the ground truth positions and the resulting tracked positions. The value of the metric is computed by calculating the Euclidean distance between each tracked position (x_{tr}, y_{tr}) and the ground truth position (x_g, y_g):

$$D_{(g,tr)} = \sqrt{\left(x_g - x_{tr}\right)^2 + \left(y_g - y_{tr}\right)^2} \qquad 8.1$$

It should be noted that this metric has a higher value when the tracked position does not match the ground truth position. Then by taking the standard deviation μ and the mean value σ of each dataset, it will be possible to identify where the algorithm is generating higher errors.

Figure 8-8 shows clustered bars which compare the standard deviation of the errors from five datasets under different conditions of illumination. It is clear from the figure that the error is higher when a shadow is cast over part of a tracked toy, whereas, the best results are achieved when using Neon source light.

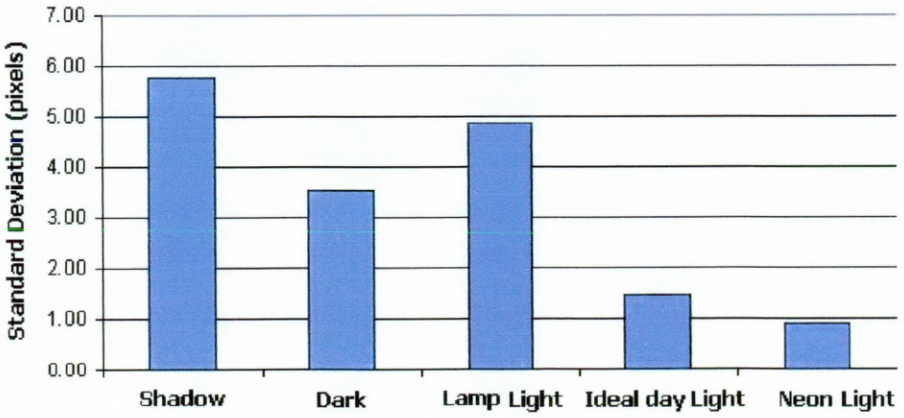
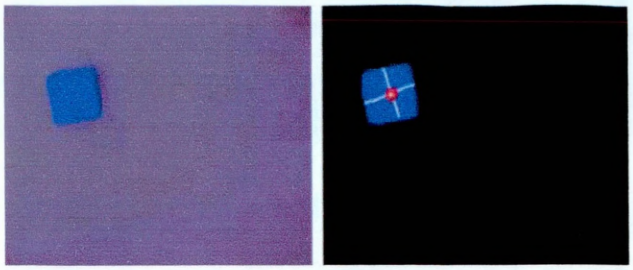
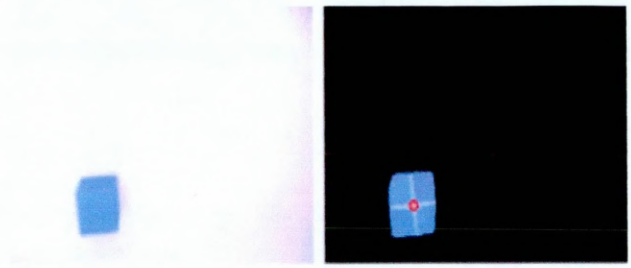


Figure 8-8: *Cluster bars showing the standard deviation of the error under five different conditions of illumination*

Figure 8-9 shows the result of the toy tracking test for two different environments, namely a dark and a bright environment. Both images show successful tracking results.



(a) Dark environment



(b) Bright environment - not direct sunlight

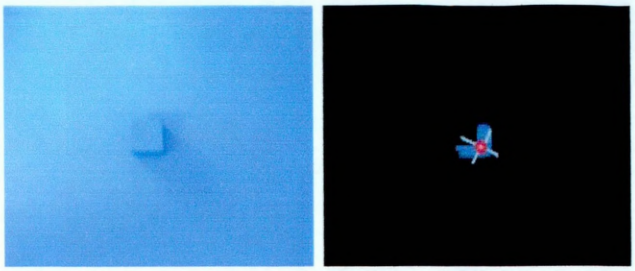
Figure 8-9: *Toy tracking under different conditions of illumination*

Table 8-1 shows the standard deviation along with the minimum and the maximum values of the error using three different backgrounds. As might be anticipated, it is noted that the error is higher when the background has the same colour as that of the toy. Similarly, the error is lower when the background provides a clear contrast with the tracked toy.

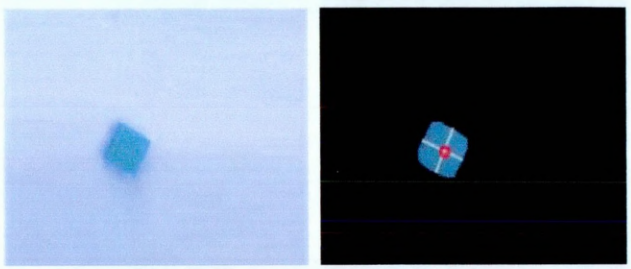
Table 8-1: *Error evaluation results (in pixels) for backgrounds of different complexity*

Test set	STDev (μ)	Max	Min
Complex background	1.46	6.00	3.16
Background of the same colour	2.49	8.00	4.47
Simple background	0.83	2.24	0.24

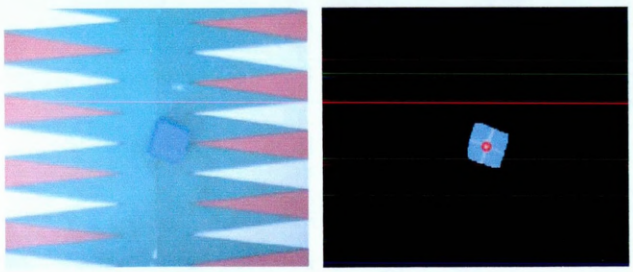
Figure 8-10 shows three pair of images from the toy tracking test using three different background complexities. The figure shows that some parts of the toys region are removed when using a background of the same colour as the toy.



(a) Blue block on blue background



(b) Blue block on a contrasting background



(c) Blue block on a complex background

Figure 8-10: *Three pairs of images from the toy tracking test for three backgrounds of different complexity*

Finally, Table 8-2 compares the average standard deviation, minimum and maximum of error values produced by each of the four testing scenarios. It is clear that using large scale toys can cause higher errors (see Figure 8-11) while error values which are caused by toy translation around the camera's field of view are the lowest (See Figure 8-12).

Table 8-2: *Error evaluation of four testing scenarios (All errors in pixels)*

Test set	STDev (μ)	Max	Min
Toy scale	5.30	15.30	0.87
Illumination changes (average error of all the illumination changes)	3.31	10.92	1.21
Background complexity	1.59	5.41	2.62
Translation	1.20	2.20	0.12

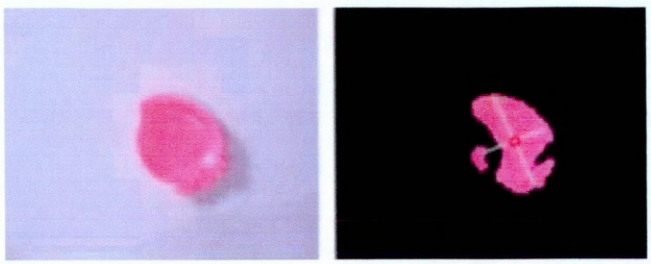


Figure 8-11: *A pair of images from the toy tracking test using a large-scale toy*

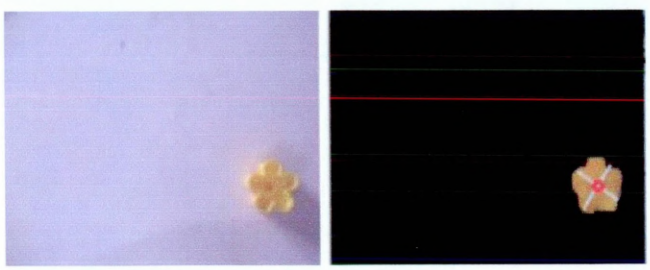


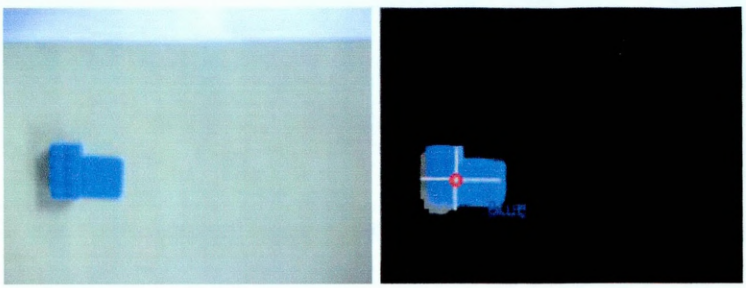
Figure 8-12: *A pair of images from the toy tracking test where the toy was tracked successfully near the edge of the image*

8.2.3 Evaluation of Labelling Connected Components

It was shown in Chapter 6 that labelling connected components is used when multiple toys are moved within a single turn. Observation of the region labelling algorithm

developed in this thesis has shown two types of errors which need to be dealt with in the future. These are:

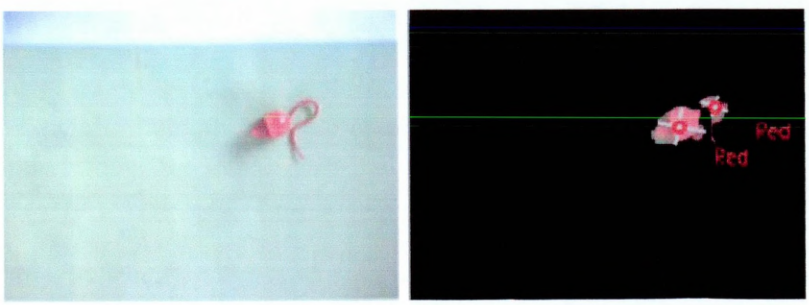
Firstly, if two toys are placed closely beside each others (less than two pixels gap) then the algorithm might give one label for two occluded toys instead of two labels. Figure 8-13 shows two toys are placed beside each and being assigned one label and one centroid instead of two.



(a) Input image with two touching bricks (b) Resulting image

Figure 8-13: *An example of labelling connected components test*

Secondly, some long and curvy toys might be segmented into multiple regions depending on the illumination. This results in the system assigning two labels to the region of the toy instead of one. Figure 8-14 shows small toy mouse with a long curvy tail. Change detection has resulted in two regions being identified, and hence two labels are assigned to the same toy.



(a) Input image of toy mouse (b) Resulting image

Figure 8-14: *An example of labelling connected components test*

8.2.4 Evaluation of Turn Detection

Skin Detection

A number of trials were carried out with the skin detection algorithm under a various illumination conditions and using various skin colours. The results of these tests have shown that the algorithm functions properly over a wide range of skin colours. Figure 8-15 shows four successful results of detecting hands of different skin colour.

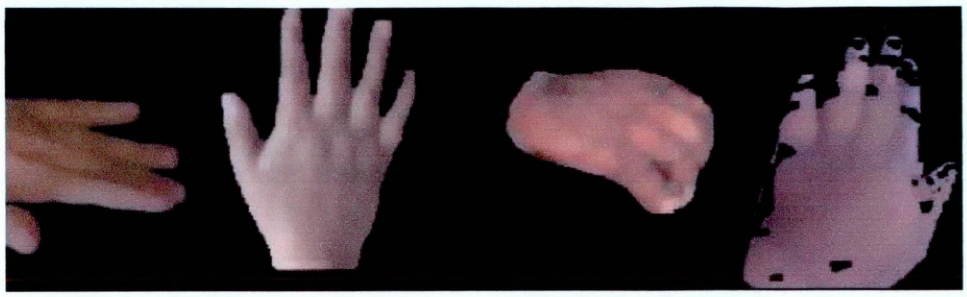
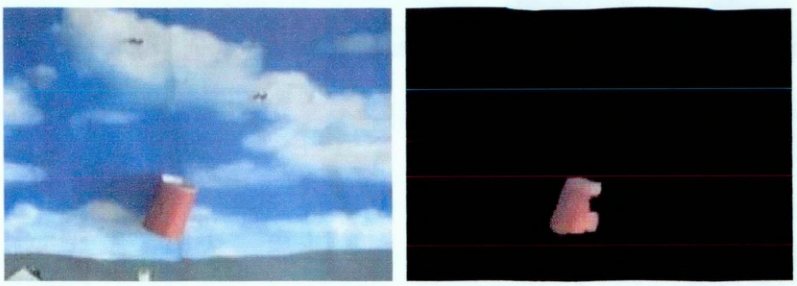


Figure 8-15: *Results of the skin detection test*

However, the skin detection algorithm has a number of limitations as it relies mainly on colour detection. For example, some objects of a generally brown colour might pass the skin detection test. These objects can be the foreground toys or regions of the background. Figure 8-16 shows an input frame image from the skin detection test along with the resulting image from the skin detection algorithm. The brown toy in the input image was partially detected as a skin object in the resulting image.



(a) Input image with a brown toy (b) Resulting image with some areas of the toy having passed the skin test

Figure 8-16: *An example of skin detection test failure because of object colour*

Figure 8-17 shows an input image with a complex background (many coloured areas including brown). The result image shows that two brown regions from the background were detected as skin regions.

(a) Input image with complex background (b) Resulting image with part of background detected as skin

Figure 8-17: *An example of skin detection test failure because of background colour*

Motion Detection

The motion detection algorithm was also tested in the turn-based interaction mode. This algorithm was found not to be susceptible to changes in illumination since the Motion History Image *MHI* contains those pixels which were in motion within the "motion period" δ and does not rely on their colour. Figure 8-18(a) shows an input image with hands moving a toy rabbit. As can be seen from Figure 8-18(b), some skin areas of the

hand did not pass the skin test, whereas the same areas appeared in the motion history image shown in Figure 8-18(c).

(a) Input image

(b) Resulting image using skin detection

(c) Resulting image using motion detection

Figure 8-18: *Invariance of motion detection algorithm*

In addition, hands or any other objects can be used to move the toys as shown in Figure 8-19.

(a) Input image with a pen used to move a toy

(b) Resulting image following motion detection

Figure 8-19: *Detection of moving objects*

8.2.5 Colour Recognition

Chapter 6 has shown that colour is a very useful cue for toy detection. This is because it is a low-level feature and computationally inexpensive. Unfortunately, colour has a serious drawback as illumination dependency which makes it difficult to work in all real world environments (Martinkauppi, 2002).

Nevertheless, an experimental work was carried out in this thesis to assess the performance of toys colour recognition under various conditions of illumination. In the tests, forty toys of four colours red, blue, green and yellow (ten toys of each colour as shown in Figure 8-20) were tested under five different conditions of illumination.

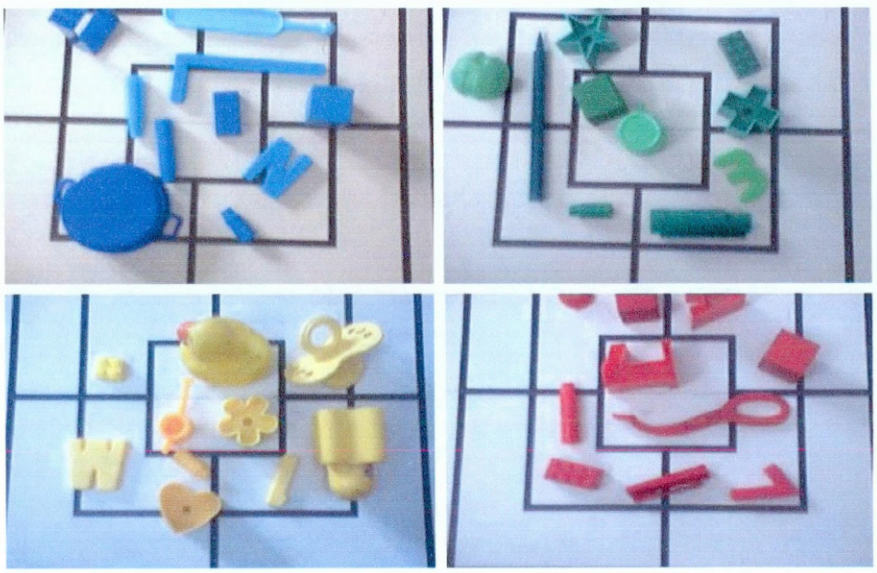


Figure 8-20: *Toys used in colour recognition evaluation tests*

These conditions were; ideal daylight, direct sun light, desktop lamp light, a Neon Light and dark environment. Three images were repeated for each illumination condition. This resulted in a total of 10 (toys) x 4 (colours) x 5 (illumination changes) x 3 (test images) = 600 images. For each of these, the input frame image was processed to create its associated output image. The resulting colours are then manually compared with the real

colours in the input images. The error rate of colour recognition for each colour is then calculated as the relationship of equation 8-2.

$$Error\ percentage = (100/N)M \qquad 8.2$$

where M is the number of test images with unsuccessful results and N is number of overall test images for each colour N=45.

Figure 8-21 shows the recognition error rates associated with each of the four colours (red, green, blue, yellow) under the different conditions of illumination. It can be seen from the graph that the error rate is higher for blue and red toys when using the lamp light source while the error rates are low when the Neon light source is used or within the highly saturated environment (dark).

In general, green was found to be generally illumination invariant while, blue can be mismatched if the illumination changed or the toy is of a low intensity blue colour.

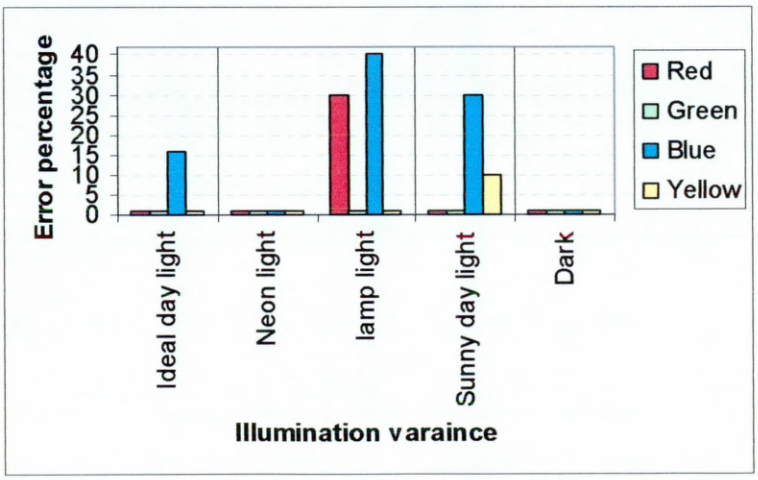
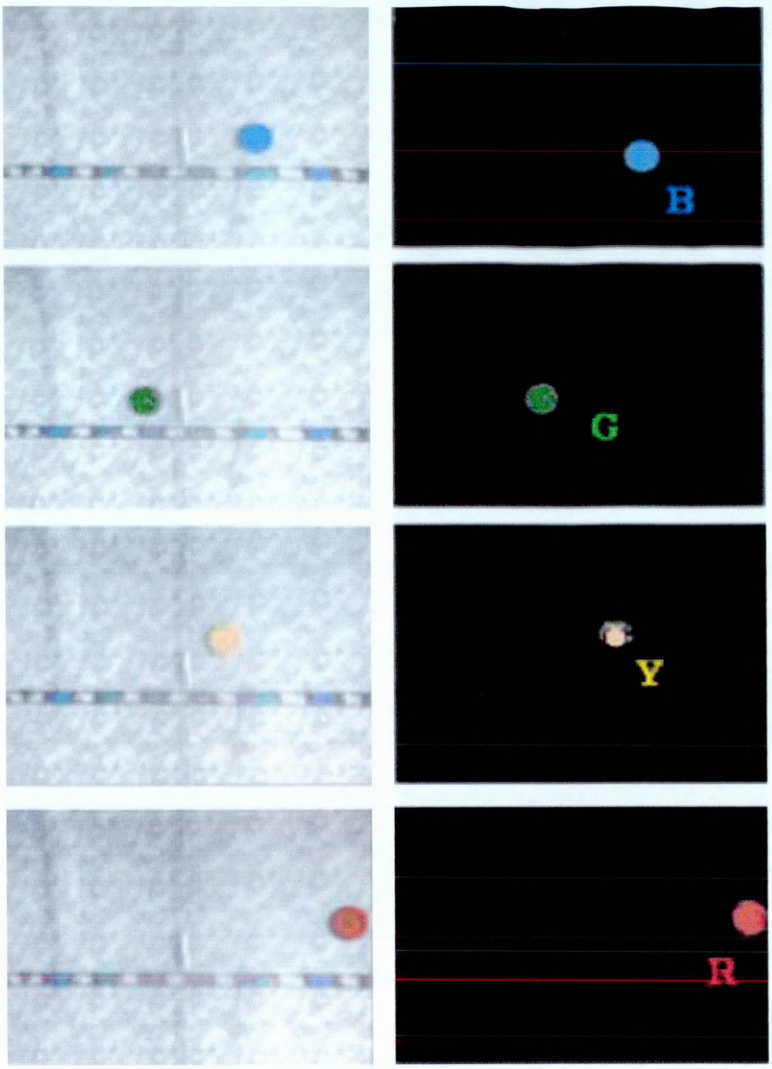


Figure 8-21: *Recognition error rates for toys of four colours (red, green, blue and yellow) under five different conditions of illumination*

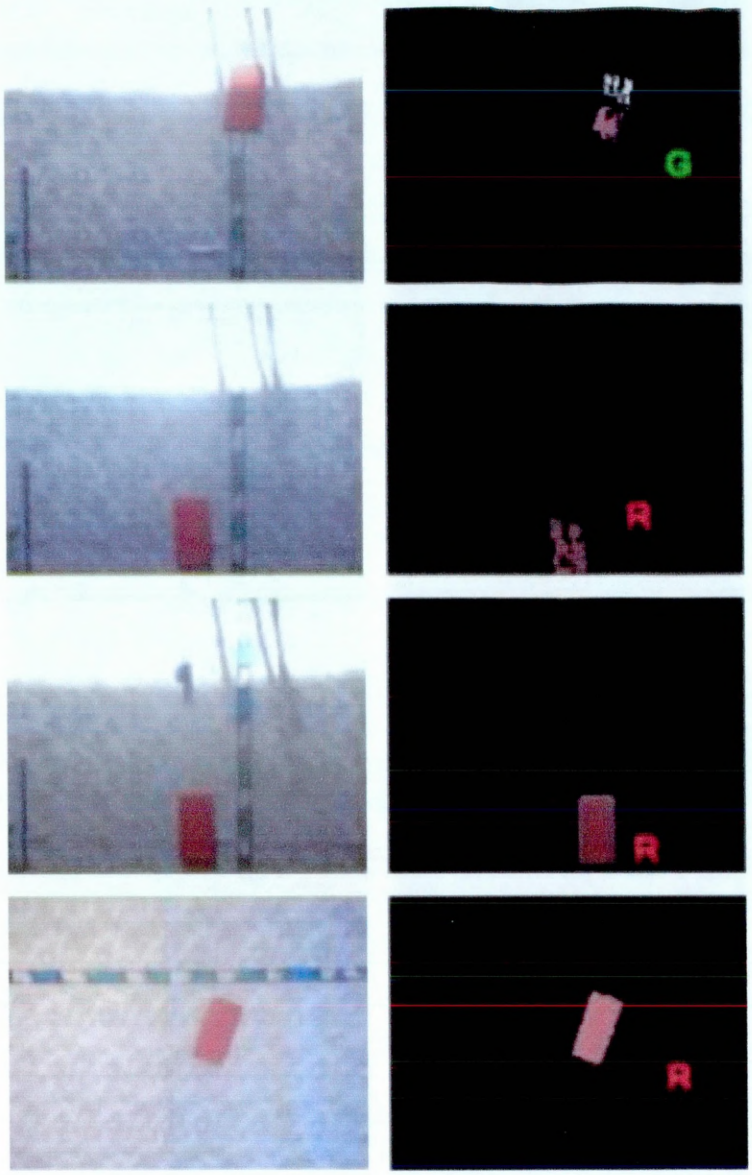
Figure 8-22 shows four images of four coloured toys (red, green, blue and yellow) along with the successful resulting images of colour recognition algorithm where each toy is classified according to its colour.



(a) Initial image (b) Result under ideal daylight

Figure 8-22: Four example of colour recognition test

Figure 8-23 shows some sample colour recognition results using sample images of a red rectangular object in various types of sunlight illumination. The first image (in the top row) shows a negative result where the red object is under direct sun light, while the other three images show that the algorithm was able to recognise the red colour successfully.



(a) Input images (b) Resulting images after processing

Figure 8-23: Four example of colour recognition test under sunlight illumination

8.3 Computational Cost

Intel's VTUNE™ Performance Analyser was used to gather timing data for the algorithms developed in this thesis. Each algorithm was run on a single-processor machine equipped with a 1.4 GHz Pentium 4 Processor for a period of two minutes. The web camera used for the tests was feeding frames at 30 fps with a resolution of 320 pixel by 240 pixel via a USB 2.0 port.

The results of testing for turn-based interaction mode and its internal functions are shown in Table 8-3 and illustrated graphically in Figure 8-24. These results demonstrate that for tracking and recognising multiple toys on a normalised panel, the total processing time consumed by the CPU is 66.41 ms per frame image. That means that this system computes the data corresponding to one frame within two frame image cycles and hence can perform effectively at 15 fps. This speed is adequate to enable practical real-time applications based on the turn-based interaction mode. In addition, it is possible to see from Table 8-3 that the "panel normalisation" and "labelling" algorithms consume the most processing time, whereas, "moments" and "motion detection" are significantly faster.

Table 8-3: *Computational time required by the turn-based algorithm and its internal functions*

Function(Algorithm)	Time per Frame Image (ms)
Moments	2.56
Motion detection	4.86
Colour recognition	11.22
Labelling	19.98
Panel normalisation	26.38
Turn-based mode	66.41

Figure 8-24 shows four bars which compare the time consumed for each interaction mode per frame image. The associated pie charts then show the time consumed by each internal function for these modes.

The figure shows that turn-based algorithm (excluding panel normalisation and region labelling) consumes 24.79 ms of CPU time per frame image. This means that turn-based algorithm can process one frame image per 25 ms when one toy is moved and without panel normalisation.

Figure 8-24 also shows that real-time-background-independent mode consumes only 1.05 ms of CPU time per frame image, which means that this mode is very fast, and can process all the frames being fed to it by the camera within one second. However, it scales with toy size, the smaller the toy the faster the algorithm is executed.

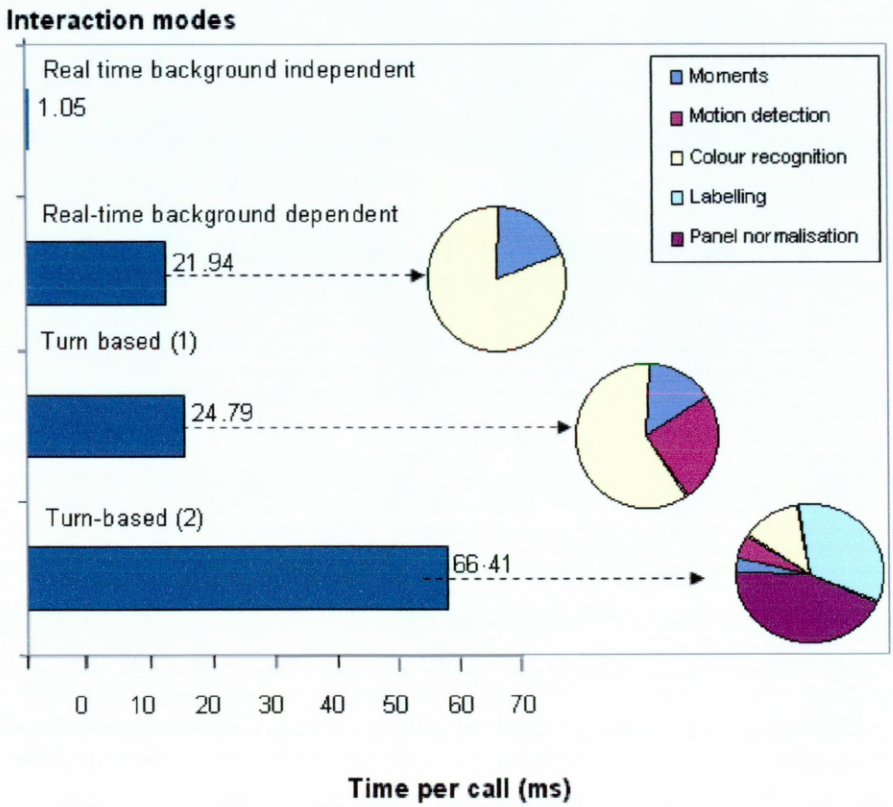


Figure 8-24: *Time consumed by each interaction mode and its internal functions per frame image*

8.4 Summary

A number of applications were designed to assess the performance of the system and subjective and quantitative tests were carried out to evaluate the error rates and the limitations of the vision algorithms developed in thesis and to measure the computational time of these algorithms. It was difficult to identify a single methodology for assessing the performance of the system and its components. This is due to the diversity in the functionality of each algorithm. For example, the results of colour recognition can be classified as positive or negative (1 or 0) while the results for toy positioning requires the calculation of the distance between the tracked position and the ground truth position. In addition, it was not practical to completely evaluate each algorithm in isolation from the others. For example, sometimes false results in segmentation might lead to spurious results in colour recognition.

However, there was an attempt to verify where the system performs better and where it fails rather than simply conducting large numbers of experiments in essentially identical conditions without any diversity. These verification experiments were carried out on small scale under changing conditions. For example, the panel normalisation presented in this thesis works well in a controlled environment but further investigation needs to be carried out to generalise its usage for a wider range of applications in non-controlled environments.

This chapter has shown that the computer vision algorithms developed in this thesis are usable and stable in the presence of normal image noise. They do not require camera calibration because all calculations take place in relation to the image plane. By tracking the four corners of the interactive panel it can be made invariant to slight motions of the camera or the background. Moreover, these vision algorithms have proved to be computationally efficient. The time consumed for each algorithm was small and suitable for real-time interaction.

The next Chapter will present some applications that were developed in this thesis to demonstrate the robustness and usability of the Interactive Toys Environment.

9 Applications

9.1 Introduction

The previous chapter presented a technical framework for evaluating the Interactive Toys Environment in terms of error rate and speed. This chapter now presents a number of applications that serve to further evaluate the Interactive Toys Environment as an enjoyable and affordable Human Computer Interface.

These applications are mainly in the field of children's playsets and computer games. The main application is the ColouredFarm playset (which is developed in this thesis) for children aged four to six and is operated in the turn-based interaction mode. Children's wish for technologies in their toys has leveraged the development of the ColouredFarm (Kanjo, 2002) as an immersive storytelling playset where children interact both with familiar toys and an animated 3-D virtual character that drives the narrative action forward.

In addition, three games will be presented using the real-time interaction mode and one game in the turn-based interaction mode.

Due to time limitations, empirical user tests are performed only for the ColouredFarm applications, further usability tests on the rest of the applications remain for future investigation.

9.2 Development and Test Framework

With the release of Intel's Computer Vision and Image Processing library (OpenCV), (see Appendix A), and the availability of the Microsoft DirectShow application programming interface (media-streaming architecture for Windows), some of the common computer vision functions are made much more accessible for computer vision programmers. These libraries implement and take advantages of specialised hardware on desktop machines (such as co-processors and MMX technology).

However, building a development framework for Interactive Toys Environment was a challenging and time-consuming task since Intel OpenCV is an open source library with limited documentation. In addition, developing programs that utilise DirectShow is not a

trivial task. The framework surrounding DirectShow is complex and not directly compatible with the OpenCV library, requiring significant time to be spent in finding out about the configuration of the OpenCV library and DirectShow.

The development and test framework for the Interactive Toys Environment has gone through three main stages:

First stage: In this stage, computer vision algorithms are built into DirectShow filters, which are separated from the user interface. These filters are the basic computer vision algorithms, which need to be connected to video capturers (e.g. cameras) and video renderers. In addition, filters have to be registered with windows after each update. A visual tool referred to as 'GraphEdit' is available in DirectShow environment to run and test the constructed filters.

Figure 9-1 shows a simple filter "ImgDiff" when the arrows in the graph depict the direction in which data is flowing. The video capture filter grabs the video from the web camera and simply passes the frames on to the filter, which passes the processed frames to the video renderer and displays the result on the computer screen.

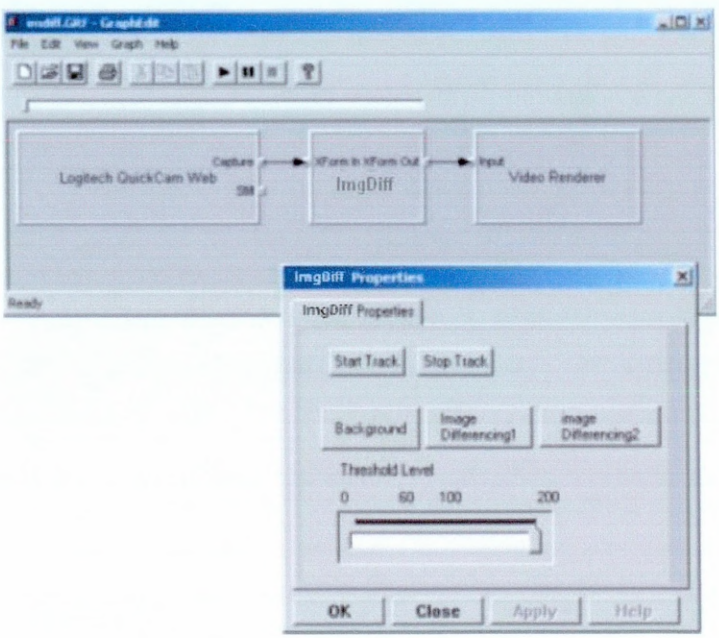


Figure 9-1: *An example of the GraphEdit interface with the ImgDiff filter and ImgDiff properties page*

Second stage: Another framework referred to as 'Camera Controller' was built to enable an Application Programming Interface (API) for reading and controlling video streams, processing frames and rendering the results as shown in Figure 9-2. Camera Controller is also used to communicate with the final user applications.

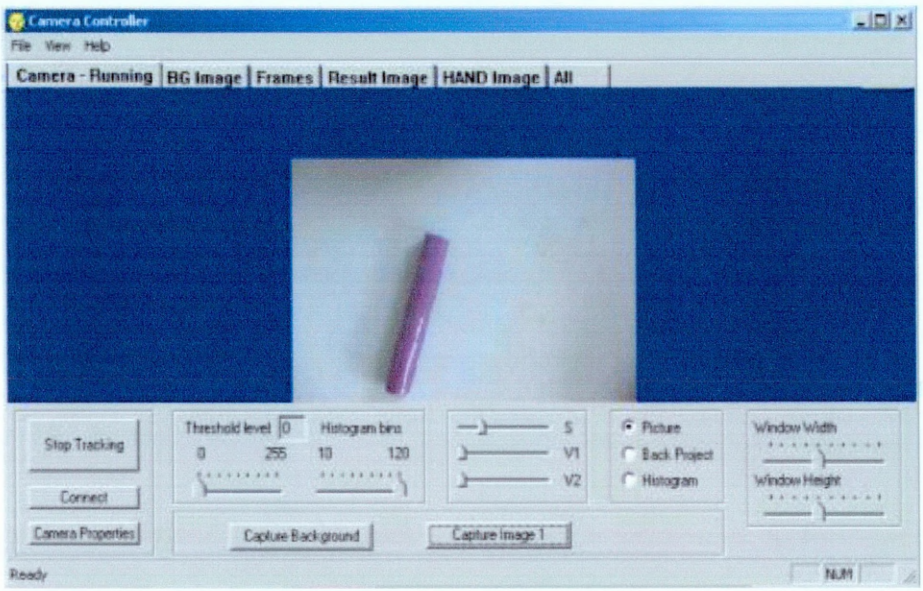


Figure 9-2: *Camera Controller interface*

Third stage: In a newer version, OpenCV has provided a universal cross-platform module for processing a video stream which bridges the gap between OpenCV and the DirectShow framework by using wrapped functions. This module is referred to as 'CvCam' and enables another interface to be constructed to support Interactive Toys development. This interface is developed in this thesis and referred to as "iToysController". iToyController provides a robust development environment and an easy link to user applications which makes computer vision development much easier without the need to register filters for every update. Figure 9-3 shows the iToysController interface while displaying the source video and the processed images (the results of the computer vision algorithms). From the figure, it can be seen that iToysController enables the user to:

- Acquire background images.
- Access camera properties.
- Switch between different interaction modes.
- Change threshold values using sliding bars.
- Display all the processed images alongside the video source.
- Connect to the final application.
- Enable the running of two cameras.

Figure 9-3: *The* iToysController *interface*

9.3 Applications for Children

Playing with toys is highly significant for children with regard to learning new skills and exploring the surrounding environments. Toys are best understood as tools of the human child. They are to train them in physical skills, to help them to develop imagination, and to enable them to explore possibilities in a creative yet safe environment. In addition,

children expect their toys to listen to them, to tell stories or may be even to come alive and have senses just like them. Clearly, there is a space to include children along with traditional playsets and computer technologies within a single framework.

In this section a number of applications of the Interactive Toys Environment developed in the course of the research are considered. These applications aim at:

- Enabling children to make the most of their senses while playing, especially by adding touch along with visual and audio activities,
- Enhancing development by providing collaborative play and hands-on tools experience and a better 3D physical overview,
- Stimulating learning and supporting knowledge integration as well as excitement.

9.3.1 Preliminary user tests on children

To help in designing some example applications in this thesis a number of user tests were carried out with children using ordinary boardgames and playsets. These tests were important in answering a number of design questions including:

- How many children can play at the same time and how many children can interact in a social manner?
- Do children follow games' rules?
- How often do children take the toys away while playing with a playset?
- Do children prefer to play their games on the table or on the floor?
- Interactive content (physical tools) is important for children, but what is the limit for adding interactive tools? In other words, do children require a lot of interactivity to enjoy playing and learn more?
- Could it be that the child will be fascinated by the use of the physical toys and forget about the main goal of using them?
- How long does it take a child to play one turn (on average)?

In the preparatory tests, children were observed while playing with five playsets/boardgames in a usability lab, which is a two-room facility (testing and observation rooms) separated by one-way glass. These tests were video-recorded using three cameras, which were fixed in three different locations.

Table 9-1 shows the age, the gender and the number of children who have participated in two usability tests.

Table 9-1: *Children participating in user tests along with their ages and gender*

	Age	Gender	Number
Test One	4	Female	2
	5	Female	1
	5	Male	1
	6	Male	2
Test Two	7	Female	1
	10	Female	1
	8	Male	1
	11	Male	1

Table 9-2 shows the board games (and playsets) used for the tests, the ages they are suitable for and the number of players.

Table 9-2: *Playsets and board games used for user tests*

Game	Ages	Number of Players
Mouse Trap	5 to 12	2-4 players
Winnie the Pooh playset	3 to 8	1-6 players
Farm Playset	4 to 10	1-6 players
Spiderman board game	8 to Adult	2-4 players
Monopoly Disney Edition	8 to Adult	2-8 players

Figure 9-4 shows three boys aged 4 to 6 playing with the "Mouse Trap" board game. It was observed that the children's hands are not always placed on the board and that each child performs a task and then withdraws their hands to give a turn to another child.

Figure 9-4: *Three boys playing with the "Mouse Trap" board game during the preliminary user tests*

Children, and in particular girls, have enjoyed playing with the farm playset (see Figure. 9-5) on a tabletop especially because there was no restrictions and rules on game play. They were able to collaborate without interfering with each other, tell stories and share their experiences about the farm animals. This experience has inspired the development of the farm application which demonstrates the power of the Interactive Toys Environment (Kanjo, 2002).

Figure 9-5: *Children interacting with the farm playset during the preliminary user tests*

9.3.2 ColouredFarm: Interactive Environment by Narrative Playmates Toys

ColouredFarm Configurations

The "Coloured Farm" playset consists of several coloured areas (see Figure 9-6), with a main playing piece named "Hadi" (see Figure 9-7) whose task is to help the family in the preparation of breakfast. In addition, there are specially tasked items such as milk and vegetables, which can be hidden either in the cupboards or in the young farmer's bag (see Table 9-3). These items were added to the game to increase the level of interactivity and to help to improve problem-solving techniques. As their role to be hidden by children (in the boxes, cupboards or in the back-bag, all of which are available in the Coloured Farm) so they do not interfere with the tracking process.

Figure 9-6: *ColouredFarm areas*

Figure 9-7: *The young farmer* (Hadi)

Table 9-3: *The coloured areas of the farm and its components*

Area name	Area colour	Toys	Special items
Family house	Cream	Father, mother, young farmer and his little brother	Cheese, wool, scarf
Dog house	Purple	Farm dog	Dog food
Chicken coups	Pink	Hens, roosters and chicks	Eggs
The pond	Blue	Ducks	Bottle of water
Cow shed	Red	Cows and calves	Bottle of milk
Sheep shed	Light green	Sheep and lamb	Ball of Wool
Pig pen	Brown	Pigs and piglets	-
The stable	Orange	Horses and foals	-
Vegetables patch	Green	Rabbits	Tomatoes, lettuce

Along the way, a narrator on the screen tells the story and guides the game. This narrator is 3-D animated character of a red rooster named "RedRooster". In addition, the computer screen can display various game views, along with speakers to output music and the sound of the animals, the RedRooster and the people in the farm. These sounds are a pre-recorded voice narration. The system uses the context established by the story for robust initialisation and performance, for example, mother farmer might call young farmer to get back to the house, and after the children take him back to the house, the system can update itself.

Children's Interaction with the ColouredFarm

Children can interact with ColouredFarm at three levels (stages) as shown in Figure 9-8:

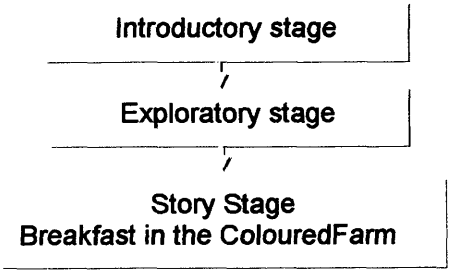


Figure 9-8: *Three levels of interaction with the* ColouredFarm

Introductory Stage: The story commences when the on screen narrator (RedRooster) gives a greeting and introduces himself to the children and then asks them to tell him their names. RedRooster then introduces the farmer's family and invites the children to explore the ColouredFarm by moving the main playing toy. When the children pick up the Hadi figure and place it in the farm house area, an conversation starts between the members of the farm family which serves to introduce the various characters.

Exploratory Stage: RedRooster then encourages children to visit the farm and find out about the different coloured areas. This exploratory stage is very important to enable the children to familiarise themselves with the environment and the use of the toys as controlling tools in the game. The children know that the environment is responsive, therefore they become curious to know what makes it different from an ordinary farm playset.

The children can move the toys around the farm arbitrarily and listen to RedRooster introducing each area. It is then up to the children to get back to the house to finish the exploratory stage. As soon as a child places Hadi back in the family's house, the family begins to converse about having a breakfast that is to be collected from various locations around the farm, and they ask Hadi to help in the preparation.

The Story Stage *"Breakfast in the Coloured Farm"*: The children move Hadi after being told where to find the milk, vegetables and meat. These are considered as special items and need to be moved in special boxes inside the farm before the start of the ColouredFarm application. The farm animals - which speak when approached - guide the children as to how to find these items and direct them to place them in Hadi's bag. When the children move Hadi, sounds and videos transform the playset into a mystical farm.

Hadi fetches vegetables, milk and eggs from the farm and while he is feeding the dog and supplying water to the horses, his mother prepares cheese from the milk and a salad from the vegetables. In addition, there are other activities; for example, getting some wool from the sheep to enable the Hadi's mother to knit a scarf.

The following is an example of the interaction between RedRooster and Hadi:

RedRooster: *"You can visit the cowshed to get some milk or the chicken to fetch some eggs"*

Hadi: *"OK, I would like to visit the cows, let's go"*.

The child moves Hadi to the cowshed area and as he approaches a video of the cows appears onto the screen. One of these cows who was grazing happily, starts to chat with Hadi:

Cow:	*"Hello young Farmer, how are you today?"*
Hadi:	*"I am fine, but I am a bit confused, my mother told me to get milk from the farm, but I do not know how?"*
Cow:	*"Oh is that all, MOO, I can help you"*
Hadi:	*"Really"*
Cow:	*"MOO, MOO, cows can help people in many different ways; we eat grass, and make milk, which we keep it for the others, this milk can be used to make cheese, yoghurt, and many, many more things...MOO"*
Hadi:	*"Wow, cows are wonderful animals"*
Cow:	*"Oh thank you young farmer for the lovely compliment. I shall have to give you a gift"*
Cow:	*"Look in my shed, there should be a big container of milk, take it to your Mummy farmer and have a lovely breakfast"*
Hadi:	*"Thank you very much, bye now"*
Cow:	*"Bye and do not forget to come back to visit me...MOO"*

Usability test on the ColouredFarm

The *Wizard of Oz* test was conducted to evaluate the ColouredFarm as an edutainment setting for children. Children of different age groups (mainly four to six) have played with the ColouredFarm. Ten children volunteered for the three tests, which were video-recorded. Different options have been tried such as two girls, two boys or a boy and two girls...etc. as set out in Table 9-4.

Each test lasted twenty to thirty minutes depending upon children's interactions with the farm. The observer was imitating the tracking technology in the other usability room. When the child moves the young farmer to the pond, the observer simulates this movement and makes a similar input to the game. Children did not see the physical and the virtual instantiations of the toy simultaneously, a 3D animated narrator speaks while the children are listening, or watching some animations, then the children move the toys while the narrator is watching.

Table 9-4: *Age, gender and number of children participated in the* Wizard of Oz *tests*

	Age	Gender	Number of children
Test One	4 to 6	Male	2
		Female	2
Test Two	6 to 9	Male	2
		Female	2
Test Three	9 to 10	Male	1
		Female	1

Figure 9-9 shows a snapshot from the Wizard of Oz test of children using the ColouredFarm. In the figure, three images from the three cameras overlooking the ColouredFarm environment for the test recording. The top right image also shows the output screen displaying the RedRooster while introducing the Hadi to the children. The other three images show two girls and two boys interacting with the ColouredFarm environment. Figure 9-9 and Figure 9-10 show more snapshots from the Wizard of Oz tests using different settings.

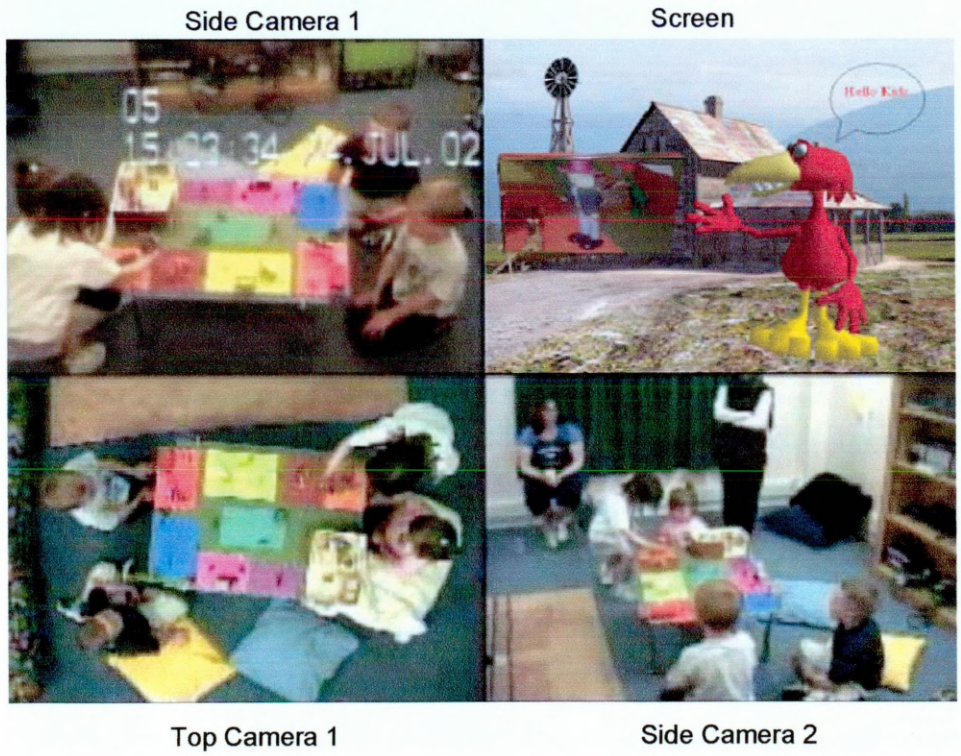


Figure 9-9: *Screenshots of the Wizard Oz test with four children (two boys and two girls)*

(a) *Two girls aged four*

(b) Two boys aged six

(c) *One girl and one boy aged ten*

Figure 9-10: *Screenshots of the Wizard Oz test*

Children were looking forward to know what makes this farm different from the traditional farm playsets. Then after playing, they were fascinated; they could not see how

did the storyteller know where they had placed Hadi! This means that the environment was giving natural responses and that the technology not visible to the children. Through observation, it was noted that children were playing interactively, and without following a particular storyline. They were consciously trying all the options and placing the toys in different places. In addition, they were able to visit the animal areas, hear stories, and learn about the benefits of the farm.

According to these observations, it was considered that ColouredFarm environment is best-suited to children aged 4 to 6. However, this application can be extended to children of different ages by incorporating storyline and dialogue that matches their needs. Following the Wizard the Oz test, further informal user tests were carried out to evaluate the performance of the fully implemented ColouredFarm application in terms of its usability and robustness. Figure 9-11 shows two children interacting with the ColouredFarm (the web camera is not visible in the figure).

Figure 9-11: *Two children interacting with the* ColouredFarm *in informal user tests*

ColouredFarm was also demonstrated to a large number of people who visited IC CAVE. Figures 9-12 and 9-13 show some of the IC CAVE visitors interacted by the Coloured Farm.

Figure 9-12: *IC CAVE visitors interacting with the* ColouredFarm

Figure 9-13: *User interacting with the* ColouredFarm

A number of users commented on the ColouredFarm being more enjoyable, engaging, and "*fun to use*" than the conventional graphical interface. In addition, user tests have supported the claim made in Chapter 1 that the Interactive Toys Environment is:

Affordable: Users were able to interact with the environment without prior knowledge of how to do so. For example, users often start by moving Hadi into different coloured areas and hence to explore the farm. Then, when they take him back to the farmhouse when the

story begins. This story guides the user as to where to go next and how to collect the items for breakfast.

Two handed interaction: Users tended to use a two-handed interaction to place the items into Hadi's bag. Usually users do not move two toys using two hands, although they can do so.

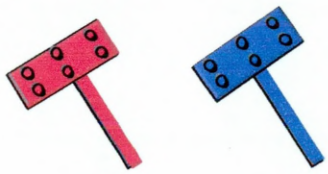
Collaborative interaction: Users can collaborate with each other while interacting with the environment. For example, in the breakfast story, one child was moving Hadi while the other child was getting some vegetables from the vegetable patch.

9.4 Computer Games applications in real-time mode

This section describes three applications developed in this thesis to demonstrate the robustness of the real-time modes of the Interactive Toys Environment in controlling games. In these applications, the user has a full control over the game by having the virtual characters in the games respond according to the direction, colour, orientation and 2-D translation vector of the physical toys.

9.4.1 "Falling Blocks" Game[13]

In this game, users are able to translate and rotate the falling blocks by moving a physical brick (as shown in Figure 9-14).



(a) Red side (b) Blue side

Figure 9-14: *Physical brick with a handle*

[13] Tetris

The user needs to do a number of actions to get the feedback from the game. These are:

- Move the toy left or right to enable the falling bricks to be translated left and right in the actual game image.
- Rotate the toy brick more than 10° clockwise to rotate the falling block right.
- Rotate the toy brick more than 10° anti-clockwise to rotate the falling block left.
- Flip the brick to the other side to get a different colour for the falling blocks.

For example, Figure 9-15 shows a screen shot of the falling blocks game while the toy brick is controlling the blue virtual falling brick.

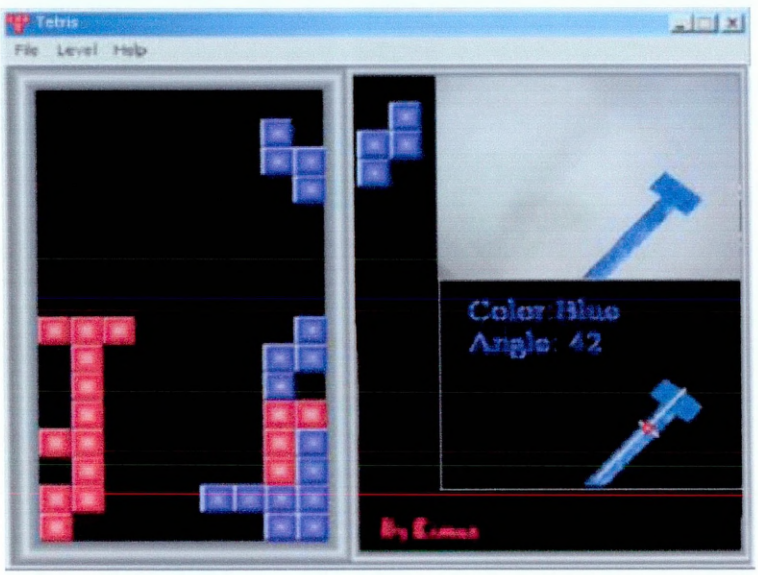


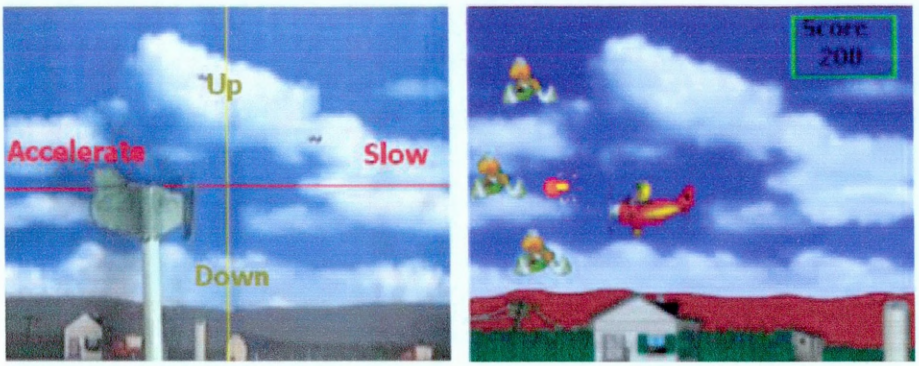
Figure 9-15: *Screenshot of Falling Blocks user interface with game actions*

9.4.2 *"Chicken Hunt" Game*

In this application a user can control a 2-D virtual plane; chase flying chickens and fire on them as shown in Figure 9-16. Similar to the Falling Blocks application, the user needs to do a number of actions as follows to get game feedback:

- Flip the toy plane to the other side to fire on flying chickens.
- Move left to accelerate the virtual plane, Move right to slow the virtual plane.

- Move up to translate the virtual plane up, move down to translate the virtual plane down.



(a) Toy plane moving on a printed background (b) Screenshot of the corresponding game actions

Figure 9-16: *The "Chicken Hunt" game*

9.4.3 *"Break" Game*

The aim in this work of developing this game application is to demonstrate the real-time-background independent mode of operation. In this game application, users can use almost any toy to control the game. The toy should be moved left and right to translate the virtual paddle left and right as shown in Figure 9-17.

Figure 9-17: *User controlling "Break" game by the movement of an ordinary pen*

Informal user tests on real-time interaction modes, and in particular the background-dependent mode, have shown that the movements of the virtual characters are very smooth since the real-time interaction mode performs at more than 15 fps as shown in Section 8.3.

9.5 Computer games in Turn-based mode:

Turn-based interaction mode was linked to the 3-D Baldur's Gate game. Every time the user moves the physical toy on the interactive panel, the detected position is translated as a control variable in the mouse queue, which moves the 3-D character accordingly on the screen. Figure 9-18 shows a user hand moving a toy on a printed scene from the game, while the screen displays the corresponding game actions. The two red circles show the positions of the physical toy and the virtual character.

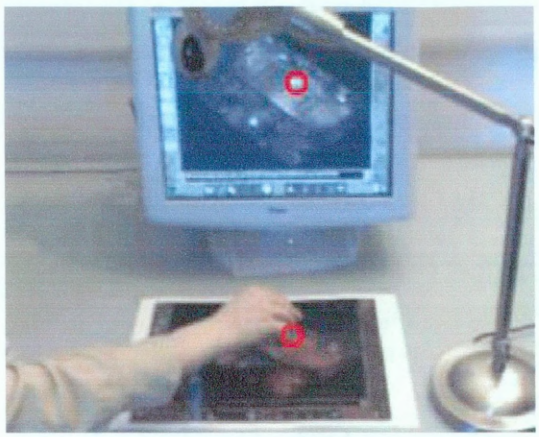


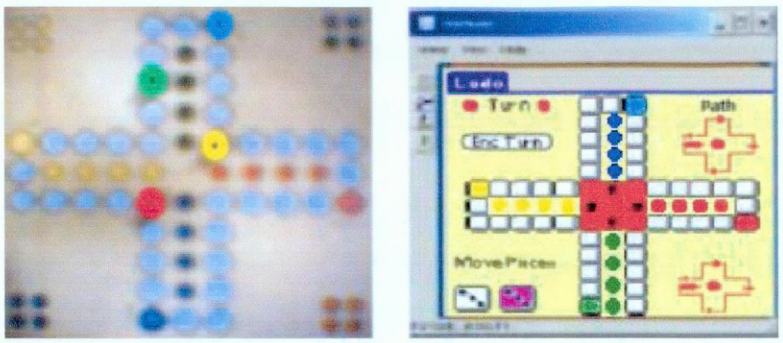
Figure 9-18: *User playing with the* Baldur's Gate *game*

9.5.1 Board Games and Robotic Applications

Most of the classic and modern boardgames have been converted in order to run on computers and games consoles. Depending on the skill of the designers and the capability of the interface, these converted games provide access to a computer-based opponent and reduce the drudgery of scoring. Unfortunately, this is usually at the expense of the simplicity of the interface presented by the more tactile original.

Therefore, it is believed that boardgames (which normally require one opponent or more) are very suitable applications for the Interactive Toys Environment in the turn-based mode. The opponent can be a human or a robot, which provides a physical feedback. However, the development of these applications is time consuming and needs a number of techniques to be incorporated in the board game rules and the robotic technology. Therefore, three prototypes were investigated in this thesis while the full implementation was left for future work. These prototypes are:

- An interface for Ludo board game, see Figure 9-19
- Incorporating a robotic system as an opposing player
- A desktop application, for example the internet messenger can be signed in to by moving a toy on a printed desktop



(a) Colour recognition image (b) Screen shot of game

Figure 9-19: *Example of the Ludo board game interface*

9.6 Summary

In this chapter, a number of applications were presented to demonstrate the effectiveness of the Interactive Toys Environment in allowing a rich but spontaneous interaction between the computer and its users. The ColouredFarm was presented as the main application, which engages children in realistic touching, listening, and learning environment. The ergonomic advantages of the ColouredFarm application, the motivation for this design and the observations of the children while they played with the ColouredFarm were explained.

The ColouredFarm started as a technological experiment, however, it is believed to be a source of inspiration for storytelling and learning. There is no mouse, keyboard, or controller to share, it helps co-operation where many children play simultaneously with no anxiety about having their turn while satisfying their wish to play.

In addition, a number of other applications in unsupervised learning and computer gaming were presented. Moreover, it is believed that an unlimited number of applications can be developed in the future in the field of robotics, boardgames, the office desktop and edutainment. The possibilities are endless. However, due to time limitations, there remain many characteristics of the Interactive Toys environment for future evaluation.

10 Closing Discussion

10.1 Thesis Summary

In this thesis, the idea of Interactive Toys Environment was introduced as an approach to advance human computer interaction by bridging the gap between the digital information and physical objects. In particular, the thesis has focused on a novel approach for realising an interactive system that allows a number of toys to be tracked and identified using a low-cost web camera on non-sensory backgrounds in order to pass the information that is represented by the toys from the physical world into the virtual world. These toys imitate real-life objects where their interactivity comes from their ability to be tracked and identified, and thus allowing the system to respond to users actions.
Interaction with the Interactive Toys Environment does not require the users to wear gloves or tagging the toys. Users can embed their own toys on a paper-printed interactive panel (with enough contrast to its background) to run various applications.

Existing computer vision methods were adopted to build novel algorithms, which enabled to building a completely novel class of applications.
After reviewing related interactive systems in **Chapter 2** and tracking technologies in **Chapter 3**, **Chapter 4** has introduced the system architecture and discussed the interaction modes: turn-based and real-time. Then panel normalisation was introduced in **Chapter 5** as an initial technique for toy position recovery in relation to the panel. It was shown that one of the most important aspects of this method is finding four corners after extracting the edges of the interactive panel which should be of a quadrangle shape(preferably rectangle) and have enough contrast to its background.
Chapter 6 discussed the use of change detection in toy tracking and demonstrated how simple combination of background subtraction and image differencing techniques could be used to enhance region segmentation; unlike many earlier authors who tackled this problem by trying to attempt to use image differencing or background subtraction on its own right. This chapter also demonstrated how a labelling connected components algorithm alongside image moments algorithm have enabled the tracking of multiple toys.
Chapter 7 discussed the use of colour as a powerful feature for toy recognition with uniform colours. Should toys of complicated colour mixture be used then another

techniques need to be incorporated in the system to perform appearance based object recognition.

In **Chapter 8** the robustness and accuracy of each individual computer vision algorithm within the system were tested. It was shown that system produced excellent results in an ideal lit environment, however some errors have occurred when the system was used in poor lit or direct sun light. In addition, the computational cost of the interaction modes and the internal algorithm were tested. These have shown that most of the algorithms are remarkably fast while retaining a reasonable accuracy and reliability.

Chapter 9 took many of the techniques and the approaches discussed in this work and applied them to several applications in particular: children playsets (i.e. ColouredFarm) and computer games. The ergonomic advantages of the proposed system, the motivation for this design and the observations of the children while they play with the ColouredFarm have been explained.

10.2 Future Work

The development of the Interactive Toys Environment can be further enhanced by both adding some interactive features and extending computer vision algorithms as follows:

10.2.1 Interactive features

A flat plasma screen can be added as an interactive panel, which provides real world experiences to the proposed applications. For example, users would be able to see small waves on a river or watch breeze moving grass while moving their toys. However, incorporating of a display screen requires an accurate synchronising between the camera image frame and the output screen frame. When the screen is displaying an image, the camera should not pass images to the computer vision algorithms.

In addition, a voice recognition engine can be incorporated in the system to enable users to talk to the computer while manipulating their toys. For example, users might like to record some stories or songs to be played back when other children interact with the same application. Furthermore, the interactive toys technology can be also combined with handheld devices using built-in cameras. This will enable users to interact with some applications using the Interactive Toys Environment on the move (e.g. in the train).

10.2.2 Computer vision

Computer vision algorithms can be extended in different areas, to include:

10.2.2.1 Shape analysis

An algorithm can be developed to detect the shapes of the toys to enable multiple toys tracking in the case of toys occlusion, which was described in Section 8.2.3. In addition, shape recognition alongside colour recognition will allow users to incorporate a free-range of toys.

10.2.2.2 Panel Normalisation

It was show in chapter 5 that panel normalisation relies on finding four corners in order to perform the mapping. These corners can be occluded by users' fingers or any other objects which might result in inaccurate mapping. Therefore, Hough Transform can be used to track the sidelines of the interactive panel instead of direct detection of the four corners.

Hough transform is believed to be computationally expensive (Chiew et al., 2004); however, it can be integrated with Kalman filter (Kalman, 1960) to reduce the computational load of line detection (Mills et al., 2003).

10.2.2.3 Colour Recognition

Toys' colours can be modelled using Gaussian mixture to enhance colour recognition algorithm.

10.2.2.4 3D Vision

It will be useful to investigate the use of two cameras to allow observing the environment from two different directions simultaneously. This can reduce the possible errors caused by the users when they move the interactive panel out of the camera's view (the camera's view will be expanded due to the use of two cameras). However, the computational time of the algorithms will be longer.

Furthermore, the use of 3D vision techniques will enable two types of advancement in the system. Firstly, a low-cost robotic response can be used to pick toys and move it according to certain regulations. Secondly, hand gesture recognition algorithms can be added to the system to enable the users to communicate with the computer using their hands gestures.

Finally, the merging of the physical and the digital worlds is made possible with the use of computer vision, which has been found to be particularly suitable for this type of applications as non-cumbering technology.

11 Reference

- Ackley J. and Dooley M., (2000), "Interactive Toys: The Child as Programmer, My robot can beat up your action figure!," In Proc. of Game Developer Conference, GDC'00, SanJose, USA, (Mar 20-24).
- Albiol A., Torres L., Bouman C.A., and Delp E. J., (2000), "A simple and efficient face detection algorithm for video database applications," In Proc. of the IEEE Intl. Conf. on Image Processing, Vacouver, Canada, vol. 2, pp. 239-242, (Sep).
- Alborzi H., Druin A., Montemayor J., Platner M., Porteous J., Sherman L., Boltman A., Taxén G., (2000), "Designing StoryRooms: Interactive Storytelling Spaces for Children Designing Interactive Systems," In Proc. of Designing Interactive Systems Symposium, DIS'00, New York City, (Aug 17-19).
- Bascle B. and Blake A., (1998), "Separability of Pose and Expression in Facial Tracking and Animation," In Proc. of the Intl. Conference on Computer Vision, ICCV'98, pp.323-328.
- Blake R., Sobel K.V., James T. W., (2004) Neural Synergy between Kinetic Vision and touch," Psychological Science, vol. 15, no. 6, pp. 397-402, (Jun).
- Bobick A., Intille S., Davis J. W., Baird F., Pinhanez C. S., Campbell L. W., Ivanov Y. A., Schutte A. and Wilson A., (1999), "The kidsroom: A perceptually-based interactive and immersive story environment," In Proc. of PRESENCE: Teleoperators and Virtual Environments, pp. 367-391, (Aug).
- Bradski G. and Davis J., (2000), "Motion Segmentation and Pose Recognition with Motion History Gradients," In proc. of IEEE WACV'00.
- Bradski G., Eruhimov V., Molinov S., Mosyagin V. and Pisarevsky V., (2001), "A Video Joystick from a Toy," In proc. of workshop on Perceptive User Interface.
- Brand J. and Mason J. S., (2000), "A comparative assessment of three approaches to pixel level human skin detection," In Proc. of Intl. Conf. Pattern Recognition, pp. 1056-1059.
- Bregler C., (1998), "Computational models of human motion," Doctoral Thesis, (Jan).
- Burt, P. J., Hong, T. H., Rosenfeld, A., (1981), "Segmentation and Estimation of Image Region Properties Through Cooperative Hierarchial Computation," In IEEE Transactions on System, Man, and Cybernetics, Vol. SMC-11, No. 12, (Dec).
- Canny J., (1986), "A Computational Approach to Edge Detection," In IEEE Transactions.
- Cassell J., Ananny M., Basu A., Bickmore T., Chong P., Mellis D, Ryokai K., Vilhjalmsson H., Smith J., and Yan H, (2000), "Shared Reality: Physical Collaboration with a Virtual Peer," In Proc. of ACM SIGCHI Conference on Human Factors in Computing Systems, Amsterdam, (Apr 4-9).
- Chai D. and Ngan K. N., (1998) ,"Locating facial region of a head-and-shoulders colour image", In proc. of IEEE Intl. Conf. on Automatic Face and Gesture Recognition, FG'98, Nara, Japan, pp. 124-129, (Apr).

- Chiew T., Hill P., Bull D. R. and Canagarajah C. N. , (2004), "Robust global motion estimation using the Hough transform for real time video coding," In Picture Coding Symposium, USA ,(15-17 Dec).
- Cipolla R. and Pentland A. (1998), "Computer Vision for Human Machine Interaction," Cambridge University Press, Cambridge, ISBN 0-521-62253-0.
- Colombo C., Del A. and Magistris, (1998), "Interacting through Visual Pointers," "Computer Vision for Human computer Interaction," In Cambridge University Press, UK, Part Two, pp. 135-138.
- Coombs D. and Brown C., (1992), "Real-Time Smooth Pursuit Tracking for a Moving Binocular Robot," In Proc. of CVPR'92, pp. 23-28.
- Courtney P. and Thacker N. A., (2001), "Performance characterisation in computer vision: The role of statistics in testing and design," In Blanc-Talon and Popescu, editors, Imaging and Vision Systems: Theory, Assessment and Applications, NOVA Science Books.
- CVOnline website, as retrieved on December 2004, from http://homepages.inf. ed.ac.uk/rbf/CVonline/,
- Davis J. and Bobick A., (1997), "The Representation and Recognition of Action Using Temporal Templates," MIT Media Lab Technical Report 402
- D'Hooge H., and Goldsmith M., (2001), "Game Design Principles for the Intel Play Me2Cam Virtual Game System," Intel Technology Journal, Issue, Three quarter.
- Dietz P., and Leigh D., (2001), "DiamondTouch: A Multi-User Touch Technology," In Proc. of UIST'01, Orlando, FLA, ACM CHI Letters, vol. 3, no. 2, pp. 219-226.
- Druin A., and Perlin K., (1994), "Immersive environments: A. physical approach to the computer interface", In Proc. of Computer Human Interaction Conference, CHI '94, Boston, MA, ACM Press, pp. 325-326, (Apr).
- Dudani S., Breeding K., and McGhee R., (1977), "Air identification by moments invariants," In proc. of IEEE Trans. on Computers, Vol. C-26, pp. 39-45, (Jan)
- Ernst H., Schäfer K., and Bruns W., (1999), "Creating Virtual Worlds with a Graspable User Interface," In Proc. of Interactions in Virtual Worlds, the Twente Workshop on Language Technology, University of Twente, pp. 45-57.
- Fitzmaurice G. W., (1996), "Graspable User Interfaces," Doctoral dissertation, Graduate Department of Computer Science University of Toronto.
- Fitzmaurice G. W., Ishii H., and Buxton W., (1995), "Bricks: Laying the foundation for Graspable User Interfaces," In Proc. Conf. of Computer Human Interaction CHI'95, ACM, pp. 422-449.
- Fjeld M., (1999), "Exploring Brick-Based Navigation and Composition in an Augmented Reality," In Proc. of HUC'99, Springer, pp. 102-116.
- Fleck, M., Forsyth, D. A., and Bregler, C. (1996). "Finding naked people". In Proc. of ECCV, vol. 2, pp. 592-602.
- Fleming D., (1996), "Powerplay: Toys as Popular Culture," Manchester University Press, NY.
- Forsyth D. A. and Ponce J., (2003), "Computer Vision: A Modern Approach," Prentice Hall, ISBN: 0-13-191193-7.

- Glasbey C.A. and Mardia K.V., (1998), "A review of image warping methods," In Journal of Applied Statistics, vol. 25, pp. 155-171.
- Gorbet M., Orth M. and Ishii H., (1998), "Triangles: Tangible Interface for Manipulation and Exploration of Digital Information Topography," In Proc. of Computer Human Interaction Conf., CHI'98, ACM, NY, USA, pp. 49-56.
- Grimson W., Stauffer R., Romano R. and Le L., (1998),"Using Adaptive Tracking to Classify and Monitor Activities in a Site," In Proc. of CVPR, Sanata Barbara, CA, pp. 22-29
- Hamadani N., (1980), "Automatic target cueing in IR imagery". In Master's thesis, Air Force Institute of Technology, WPAFB, Ohio, (Dec).
- Haralick R. M., (1981), "Some neighbourhood operations," In Real Time/Parallel Computing Image Analysis (M. Onoe, K. Preston, and A. Rosenfeld, Eds.), Plenum Press, New York.
- Harris C.G. and Stephens M., (1988), "A combined corner and edge detector," In 4th Alvey Vision Conference, pp. 147-151.
- Hirakawa M. and Hewagamage K. P., (2001),"Situated Computing: A Paradigm for Mobile User Interaction with Multimedia Sources," In Special Volume on Multimedia Software Engineering, Annals of Software Engineering, vol. 12.
- Holmquist L.E., Redström J. and Ljungstrand P., (1999), "Token-Based Access to Digital Information," In Proc. of HUC'99, Springer, pp. 234-245.
- Hough, P. V. C., (1962), "Methods and Means for Recognising Complex Patterns," U.S. Patent 3.069,654.
- Hu M. K. (1962), "Visual pattern recognition by moment invariants," In Proc. of IRE Transactions on Information Theory, Vol. IT-8, pp. 179-187.
- Huang A.C., Ling, B.C. Ponnekanti, S. and Fox, A., (1999), "Pervasive Computing: What Is It Good For?" In Proc. of the ACM Intl. workshop on Data engineering for wireless and mobile access. New York, NY, USA. ACM Press. pp 84-91.
- Huttenlocher D. P., Noh J. J., and Rucklidge W. J., (1993)," Tracking Non-rigid Objects in Complex Scenes," In Proc. of Intl. Conference for Computer Vision ICCV'93, Berlin, vol. 12, pp. 93-101, (May).
- Ishii, H. and Ullmer, B., (1997), "Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms," In Proc. of Conf. on Human Factors in Computing Systems (CHI '97), Atlanta, ACM Press, pp. 234-241, (Mar).
- Jacob R. J. K., Ishii H., Pangaro G., and Patten J., (2002), "A Tangible Interface for Organizing Information Using a Grid," In Proc. of Computer Human Interaction Conference, CHI'02, Minneapolis, Minnesota, USA, vol. 4, no. 1, pp. 339-346, (April 20-25).
- Johnson D. M., (1984), "Children's Toys and Books: Choosing the Best for All Ages from Infancy to Adolescence," Scribner, NY.
- Jones M. and Rehg J., (1999), "Statistical Color Models with Application to Skin Detection," In proc. of CVPR'99, pp. 274-280.
- Jordao L., Perrone, M., Costeira, J., and Santos-Victor, J., (1999), "Active face and feature tracking", In Proc. of Intl. Conf. on Image Analysis and Processing, pp. 572-577.

- KaewTraKulPong P. and Bowden R., (2001), "An improved adaptive background mixture model for real-time tracking with shadow detection," In Proc. of AVBS'01, (Sep).
- Kalman R. E., (1960),"A new approach to linear filtering and prediction problems," Trans of the ASME, Journal of Basic Engineering, Ser D, V. 82, pp. 35-45.
- Kanjo E. and Astheimer P., (2002), "Colour Farm: Interactive Toys Environment for Storytelling and Games Applications," In Proc. of 8th Intl. Conf. on Virtual Systems and Multimedia, VSSM'02. Gyeongju, Korea, (Sep).
- Kanjo E. and Astheimer P., (2002), "Interactive Environment by Narrative Playmates Toys," In Proc. of ACM SIGGROUP Bulletin, Vol. 23, No. 2, pp. 6-7.
- Lowe D. G., (1992), "Robust Model-Based Motion Tracking through the Integration of Search and Estimation," In Proc. of Intl Journal of Computer Vision, IJCV, vol. 8, no. 2, pp. 113-122, (Aug).
- Lumia R., Shapiro L., and Zuniga O., (1983), "A New Connected Components Algorithm for Virtual Memory Computers," Computer Vision, Graphics, and Image Processing, vol. 22, pp. 287-300.
- Mackay W. E. and Fayard A. L., (1998), "Designing Interactive Paper: Lessons from Three Augmented Reality Projects," In Proc. of IWAR'98, Natick, Massachusetts, pp. 81-90.
- Maggioni C., (1993), "A novel gestural input device for virtual reality," In Proc. of IEEE Annual Virtual Reality Intl. Symposium, pp. 118-124.
- Mandryk R. L., Maranan, D. S., Inkpen, K. M., (2002), "False Prophets: Exploring Hybrid Board/Video Games," In Proc. of Computer Human Interaction Conference, CHI'02. ACM Press
- Marks R., (2001), "Using Video Input for Games," In Proc. of Intl. Conf. Game Developer, San Joes, CA, March, pp. 20-24.
- Martinkauppi B., (2002), "Face Colour under Varying Illumination - Analysis and Applications," PhD dissertation in the university of Oulu, Finland.
- Mathworks, "Edge Detection," In www.mathworks.com, retrieved (July, 2004).
- Mazalek A., Davenport G., Ishii H., (2002), "Tangible viewpoints: a physical approach to multimedia stories", In ACM Multimedia, pp. 153-160.
- Metoyer R. A. and Hodgins, J. K., (2000), "Animating Athletic Motion Planning by Example," In Proc. of Graphics Interface 2000, Montreal, Quebec, Canada, pp.61-68, May 15-17.
- Michahelles F. and Schiele B., (2003), "Sensing Opportunities for Physical Interaction", In Proc. of Workshop on Real World User Interfaces", a workshop at the Mobile HCI Conference, Udine, Italy, (Sept 8).
- Mills S., PridMore T. and Hills M., (2003), "Tracking in a Hough Space with the Extended Kalman Filter," In Proc. of British Machine Vision Conference, BMVC'03, Norwich, pp. 172-180.
- Morries T., (2004), "Computer Vision and Image Processing", ISBN Palgrave Macmillan 0-333-99451-5.
- Norman D.A., (1988). "The Psychology of Everyday Things," New York: Basic Books.

- Oka K., Sato Y. and Koike K., (2002), "Real-time tracking of multiple fingertips and gesture recognition for augmented desk interface systems," In Proc. IEEE Int'l Conf. of Automatic Face and Gesture Recognition, FG '02.
- OpenCV reference manual, as retrieved on December 2004, from www.cs.unc.edu/ Research/stc/ FAQs/OpenCV/OpenCVReferenceManual.pdf.
- Patten J., Ishii, H., Hines, J., and Pangaro G., (2001), "A wireless object tracking platform for tangible user interfaces," In Proc. of ACM Computer Human Interaction Conference, CHI'01, pp. 253-260.
- Rauterberg M., Fjeld M., Krueger H., Bichsel M., Leonhardt, U, Meier M., (1997), "BUILD-IT: A Computer Vision-based Interaction Technique for a Planning Tool," In Proc. of BCS-HCI conference, pp. 303-314.
- Ritter G. X. and Wilson J. N., (2001), "HANDBOOK OF second edition Computer Vision Algorithms in Image Algebra," CRC Press.
- Rosin P., (1997), "Thresholding for Change Detection," In Proc. of, British Machine Vision Conference, BMVC'97.
- Rosin P.L. and Ellis T. , (1995), "Image difference threshold strategies and shadow detection," In proc of British Machine Vision Conf, pp. 347-356.
- Ryokai K. and Cassell, J., (1999), "StoryMat: A Play Space with Narrative Memory," In Proc. of IUI '99, ACM.
- Sato, Y., Kobayashi, Y. and Koike, H., (2000), "Fast Tracking of Hands and Fingertips in Infrared Images for Augmented Desk Interface," In Proc. of the Intl Conf. on Automatic Face and Gesture Recognition, Grenoble.
- Saxe D. and Foulds R., (1996), "Towards robust skin identification in video images," In Proc. of the Second Intl. Conf. on Automatic Face and Gesture Recognition, Killinton, Vermont, pp. 379–384, (Oct).
- Scott S. D., Grant, K. D. and Mandryk, R. L., (2003), "System Guidelines for Co-located, Collaborative Work on a Tabletop Display," In Proc. of European Conf. Computer-Supported Cooperative Work, ECSCW'03, (Sep).
- Sharpio L. and Stockman G., (2001), "Computer Vision," Prentice Hall Upper Saddle River, NJ 07458.
- Shneiderman B., (1983), "Direct Manipulation: a step beyond programming languages," In IEEE Computer, vol. 16, pp. 57-69.
- Shwe H. and Francetic A., (2000), "Technology-Enhanced Play: Smarter Play for Smarter Toys," In Proc. of Games Developers Conference.
- Smith S. M. and Brady J. M., (1997), "SUSAN a new approach to low level," In Intl. Journal of image processing, Intl. Journal of Computer Vision, Vol. 23, No.1, pp. 45-78
- Sobottka K. and Pitas I., (1996), "Face localization and facial feature extraction based on shape and color information," In Proc. of the IEEE Intl. Conf. on Image Processing, Lausanne, Switzerland, September, vol. 3, pp. 236–241.
- Sugimoto M., Kusunoki, F., and Hashizume, H., (2000), "ePro: A System for Supproting Collaboration that Enhances Interactions," In Proc. of IEEE Systems, Man, and Cybernetics Conference, (SMC2000), Nashville, TN, pp. 745-750.

- Sutcliffe A., (1995), "Human-Computer Interface Design," 2nd Edition, Macmillan, London, ISBN 0-333-59499-1, pp. 168-174.
- Terrillon J.C., David M. and Akamatsu S., (1998), "Automatic Detection of Human Faces in Natural Scene Images by use of a Skin Color Model and of Invariant Moments", In Proc. of the third Intl. Conf. on automatic face and gesture recognition,. Nara, Japan, pp. 112-117.
- Ullmer B. and Ishii H., (1997), "The MetaDesk: Models and Prototypes for Tangible User Interfaces," In Proc. of User Interface Software and Technology, pp. 223-32, (Mar).
- Ullmer B., (2002), "Tangible Interfaces for Manipulating Aggregates of Digital Information," PhD dissertation, MIT Media Lab.
- Umbaugh S. E., (1999), "Computer Vision and Image Processing," Pretenic Hall Upper Saddle River, New Jersey 07458, Chapter 2 "Image analysis".
- Vildjiounaite E., Malm E., Kaartinen J., and Alahuhta P., (2002), "Location Estimation Indoors by Means of Small Computing Power Devices, Accelerometers, Magnetic Sensors and Map Knowledge," In Pervasive, Zurich, pp. 211–224.
- Wang H. and Brady J.M. (1992), "Corner detection with subpixel accuracy," University of Oxford, Dept. of Engineering Science Technical Report: OUEL 1925/92.
- Wang H. and Chang S. F., (1997), "A highly efficient system for automatic face region detection in mpeg video," In IEEE Transactions on circuits and system for video technology, vol. 7, no. 4, pp. 615–628, (Aug).
- Want R., Fishkin K. P., Gujar A., Harrison B. L., (1999), "Bridging Physical and Virtual Worlds with Electronic Tags," In proc. of Computer Human Interaction Conference, CHI'99.
- Weiser M., (1991), "The computer for the twenty-first century," Scientific American, pp. 94-104.
- Wellner P., (1993), "Interacting with Paper on the Digital Desk," In Communications of the ACM, vol. 36, no. 7, pp. 87-96.
- Wellner P., (1993),"Self Calibration on the Digital Desk," In Euro PARC Technical Report EPC-93-109.
- Yacoob Y., Davis L., Black M., Gavrila D., Horprasert T. and Morimoto, (1998), "Looking at People in Action", In "Computer Vision for Human computer Interaction", Cambridge University Press, UK, Part Two, pp. 155-162.
- Yang M. H. and Ahuja N., (1998), "Detecting human faces in colour images", In Proc. of the Intl. Conf. on Image Processing, Chicago, IL, pp. 127-130, (Oct 4-7).
- Yang Y. H. and Levine M. D., (1992), "The Background Primal Sketch: An Approach For Tracking Moving Objects," In Proc. of Machine Vision and Applications, MVA'92, Vol. 5, pp. 17-34.
- Zarit B. D., Super B. J., and Quek K. H., (1999), "Comparison of Five Color Models in Skin Pixel Classification," In proc. of Intl. Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-time Systems, ICCV'99, pp. 58-63, (Sep).

- Zhang Z. (2003), "Vision-based Interaction with Fingers and Papers," In Proc. of Intl. Symposium on the CREST Digital Archiving Project Tokyo, Japan, pp. 83-106, (May 23–24).
- Zucker S., (1976), "Region Growing: Childhood and Adolescence," In proc. of Computer Graphics and Image Processing, vol. 5, pp. 382-399.

12 Appendix A: Computer Vision Software and Tools

In the next sections, a brief description of the main investigated software relevant to computer vision is presented.

12.1 KBVision

A commercial product, KBVision is one of the few packages intended to be used for problems in image understanding. KBVision contains visual programming tools, data visualization tools, libraries of routines for accessing files and interfaces to graphics functions. However, KBVision also supports intermediate symbolic representations (e.g. lines, regions and polygons) and allows the user to manipulate them in much the same way as images. The low-level software is written in C and the higher-level software for performing reasoning about image content is Lisp-based.

12.2 Matlab

MATLAB integrates mathematical computing, visualization, and a programming language to provide a flexible environment for technical computing. It also has an adequate image processing and image acquisition toolbox as well as neural network, Signal Processing, and many other useful toolboxes.
However, it consumes a significant amount of time to process images and deal with data which makes it not practical for real-time applications.

12.3 Open CV

In the year 1999, Intel Corporation developed a universal toolbox for research and development in the field of Computer Vision and Image Processing. The OpenCV Library is a way of establishing an open source vision community that makes better use of the latest computer vision techniques in computing environments. OpenCV software provides a set of image capturing, image processing, pattern recognition functions with platform-independent interface and supplied with whole C source.

The 2003 release of OpenCV includes an optional interface that all OpenCV functions can be imported into Matlab. In addition, OpenCV can be used together with many other libraries that are developed by Intel.
These libraries are:

12.3.1 Image Processing Library (IPL)

The Intel Image Processing Library provides a set of low-level image manipulation techniques in standard DLLs and static libraries form. The library contains functions that perform filtering, thresholding, and transforms (e.g. FFT, DCT, Geometric), as well as arithmetic and morphological operations.

12.3.2 Portable Video Capture Library (CvCAM)

CvCam is distributed as a part of Intel's OpenCV library, which uses some functionality of OpenCV. CvCam is a universal cross-platform module for processing video stream from video cameras. It is implemented as a dynamic link library (DLL) for Windows and as a shared object library for Linux systems and provides a simple and convenient Application Programming Interface (API) for reading and controlling frames from a video stream and rendering the results.

12.3.3 Portable GUI library (HighGUI)

HighGUI supports the functionality of OpenCV Library and it can be used to:

- Read/Write images in several formats (e.g. BMP, JPEZ and TIFF)
- Create windows and display images. (HighGUI remember window's content, and no need to implement repainting callbacks)
- Simple interaction facilities: trackbars, getting input from keyboard and mouse

13 Appendix B: Pixel Connectivity

The notation of pixel connectivity describes a relation between two or more pixels. For two pixels to be connected they have to fulfill certain conditions on the pixel brightness and spatial adjacency.

To formulate the adjacency criterion for connectivity, the notion of neighbourhood need to be introduced. For a pixel p with the coordinates (x, y) the set of pixels given by:

$$N_4(p) = \{(x+1, y), (x-1, y), (x, y+1), (x, y-1)\}$$

is called its 4-neighbours, These pixels are shown in Figure 13-1.

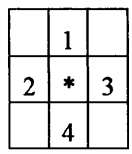


Figure 13-1: *Four neighbour pixels*

Its 8-neighbours are defined as:

$$N_8(p) = N_4(p) = \{(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)\}$$

These pixels are shown in Figure 13-2:

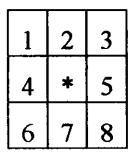


Figure 13-2: *Eight neighbour pixels*

From this, the definition for 4- and 8-connectivity can be inferred as follows:

Two pixels *p* and *q*, both are *4*-connected if *q* is from the set $N_4(p)$ and 8-connected if *q* is from $N_d(p)$.

A set of pixels in an image, which are all connected to each other is called a ***connected component***. Finding all connected components in an image and marking each of them with a distinctive label is called connected component labelling as explained in Section.

14 Appendix C: Comments on Camera test

A number of web-cameras was tried and tested using real-time and turn-based modes under different lighting conditions. Table 15-1 shows a number of tested web cameras with their resolution and frame rate per second.

The best performance was obtained from Logitech Pro 4000 for several reasons: The camera's auto exposure performance is notable (comparing to other cameras such as Lego Logitech camera). When going from dim to bright light, the camera recover quickly and with a minimum of exposure bounce, providing better results for video recording. The VGA image sensor and the lens produce generally sharp photos. The automatic white balance gives consistent colour between different light sources. Logitech has reduced the default colour-saturation level, which more accurately reproduces colours. The distortion according to the manufacturer is less than 5% of axial at corners, though the angle of view has been reduced slightly (40°). This was proven in Section 8.2.2 where no higher error occurred at the image edges and corners.

Table 14-1 *List of evaluated Web Cameras*

Web Camera		Resolution	Frame per Second
	Lego Logitech Web Cam	352 x 288	30
	Intel Me2WebCam	180x 120	15
	Logitech Web Cam	640 x 480	30
	Creative Web Cam	640 x 480	30
	SiPix web3 digital/ Web Cam	640x480	15
	Sony Web Cam Ez kit	640 x 480	25
	Philips Toucam Fun USB PCVC730K	640x480	30
	Logitech Quick Cam Pro 4000	640x480	30
	The Creative Web Cam PRO eX	640x480	30

On the other hands, a set of tests were carried out using two cameras. One camera was set with top-down view images and the other was set to view side images as shown in Figures 14-2 (a) and (b).The top-down camera was running toys tracking and recognition, motion and skin detection while the side camera was running toys tracking and recognition only. The two web cameras did not work on USB 1.0 because of the limited bandwidth. In addition, some web cameras did not function properly when connected to the computer simultaneously with other USB devices. Using two cameras 3-D vision is interesting step toward more intelligent environment however; more work in this field is left for future work.

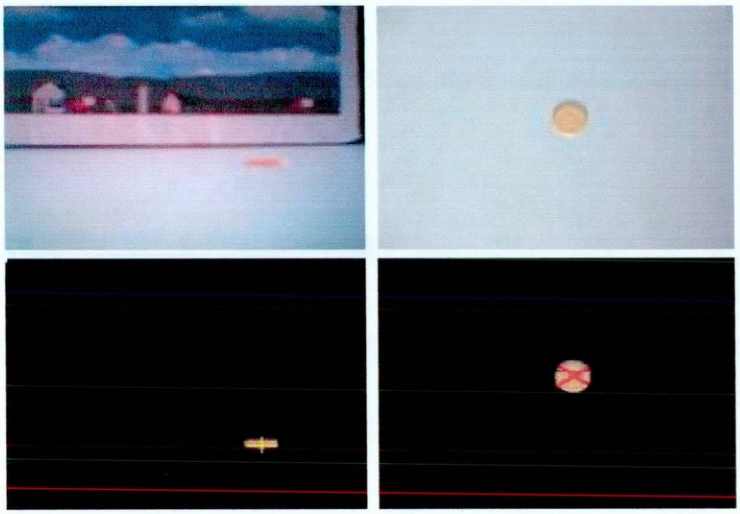


Figure 14-1: *Example of toys tracking using two cameras (a) Side camera, (b) Top-down camera*